VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2016 Martin Krátký

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

SIMULACE PLC V NÁSTROJI TWINCAT
PLC SIMULATION USING TWINCAT TOOL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Martin Krátký**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jan Mašek**
SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Martin Krátký ***ID:*** 154662
Ročník: 3 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Simulace PLC v nástroji TwinCat

POKYNY PRO VYPRACOVÁNÍ:

Prozkoumejte možnosti nástroje TwinCat pro programování programovatelného logického automatu (PLC). V rámci práce bude popsán princip PLC, dále bude vytvořena simulace robotické ruky v nástroji TwinCat, která bude řízena s pomocí aplikace s grafickým rozhraním vytvořené v jazyce JAVA. Výsledky simulace budou vykresleny do grafů.

DOPORUČENÁ LITERATURA:

[1] Z. HUA, J. WU, K.D. MUELLER-GLASER a O. SIMON. A Noise Analysis Based Channel Coding Technique for Multicarrier Channel of an Industrial PLC System. In: 2006 IEEE International Symposium on Power Line Communications and Its Applications [online]. 2006 [cit. 2015-10-19]. DOI: 10.1109/isplc.2006.247431.

[2] ANKIT DUBEY a RANJAN K. MALLIK. PLC System Performance With AF Relaying. IEEE Transactions on Communications [online]. 2015, 63(6): 2337-2345 [cit. 2015-10-19]. DOI: 10.1109/tcomm.2015.2427171.

Termín zadání: 1.2.2016 ***Termín odevzdání:*** 1.6.2016

Vedoucí práce: Ing. Jan Mašek
Konzultant bakalářské práce:

doc. Ing. Jiří Mišurec, CSc., *předseda oborové rady*

ABSTRAKT

Bakalářská práce je o možnostech nástroje TwinCat pro programování programovatelného logického automatu (PLC). V rámci práce bude popsán princip PLC a využití PLC v dnešní době. Dalším bodem v práci bude popsání vývojového nástroje TwinCAT a přehled jeho programových jazyků danou normou IEC/EN 61131-3.
Hlavním přínosem práce bude vytvoření a popis programu ve vývojovém nástroji TwinCAT ve verzi 2. Hlavním výsledkem pak bude simulace robotické ruky vytvořené v programu TwinCAT ve verzi 3 a pro tuto simulaci také bude vytvořeno grafické rozhraní navržené v programu Visual Studio v jazyce C#.

KLÍČOVÁ SLOVA

Hardware, Programové organizační jednotky (POU), Programovatelný logický automat (PLC), Software, TwinCat, Visual Studio

ABSTRACT

Bachelor project is about options in TwinCat 2 for programming the programmable logic controller (PLC). As part of the work will be described the principle of the PLC and PLC uses nowadays. The principe of developing tool TwinCAT will be described and an overview of the programming languages that standard IEC / EN 61131-3.
The main contribution of this work is to create a program description in TwinCAT development tool in version 2. The main result will be a simulation of robotic arm created in TwinCAT version 3 and for this simulation will also be created graphical interface in Visual Studio in language C #.

KEYWORDS

Hardware, Program Organization Units (POU), Programmable Logic Controller (PLC), Software, TwinCate, Visual Studio

KRÁTKÝ, Martin *Simulace PLC v nástroji TwinCat*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 59 s. Vedoucí práce byl Ing. Jan Mašek

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Simulace PLC v nástroji TwinCat“ jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor(ka) uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Janu Maškovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic
http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .
podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Programovatelné automaty jsou v dnešní době velmi rozšířené v průmyslu automatizace. Vznikly z důvodu nedostatku klasických mikropočítačů, které neměly výstupní signály přizpůsobené potřebám technologie proto se začaly vyvíjet specializované mikropočítače, u kterých byly technické prostředky a programové vybavení určeny především pro řízení technologických procesů. Tyto mikropočítače se začaly nazývat programovatelnými automaty (PLC).

PLC jsou tedy mikropočítače, které jsou speciálně přizpůsobeny svým hardwarem a softwarem pro řízení technologických procesů logického, resp. kombinovaného typu (analogového a logického).

V dnešní době dokáží PLC provádět kromě základních logických funkcí i matematické operace, přesuny bloků dat, zpracovávat spojité signály. Dnes již najdeme spoustu vývojových nástrojů pro programovaní PLC. Každá firma vytvářející PLC má svůj vlastní vývojový nástroj. Ale i přes to, že jsou to firmy fungující v rozdílných zemích mají přijatou společnou směrnici, normu IEC 61131-3, která představuje první vážný pokus o standardizaci programovacích jazyků pro průmyslovou automatizaci.

Pro tuto práci byl vytvořen program pro PLC, který řídí pohyby vrtačky vertikálním i horizontálním směrem, a také byl vytvořen program pro řízení robotické ruky s grafickým rozhraním které bylo vytvořeno ve Visual Studiu v jazyce C#.

Grafické rozhraní bylo vytvořeno v jazyce C# namísto v jazyce Java z důvodu technické problematiky komunikace Javy a programu TwinCAT. Program TwinCAT ve verzi 3 je začleněn do programu Visual Studio, ve kterém je zaveden jako nové vývojové prostředí. Snaha o navázání spojení mezi Javou a TwinCATem proběhlo neúspěšně z neznámých důvodů. Mezi těmito důvody mohla být špatná platforma (zkoušeno ve virtuálním prostředí Windows XP, novější Windows nebyl použit z důvodu nefunkčnosti TwinCATu ve verzi 3), špatný procesor na zkoušeném počítači (AMD) nebo špatná komunikace mezi Eclipsem (programový nástroj pro programování Javy) a Visual studiem.

Hlavním přínosem práce je program pro PLC vytvořený v programovém prostředí TwinCAT od německé společnosti Beckhoff, který se stará o řízení robotické ruky. Dalším přínosem práce je grafické rozhraní sloužící pro ovládání programu robotická ruka vytvořeného v prostředí TwinCAT, toto rozhraní je vytvořeno v jazyce C#. Při užití TwinCATu se dá využít jeho schopnost virtuální simulace PLC pro zjednodušené programování bez nutnosti vlastnit reálný programový automat. Výsledky práce jsou grafy vytvořeny v programu TwinCAT Scope view popisující práci programu robotická ruka, když tento program je ovládán přes grafické rozhraní typu manuální ovládání nebo automatické ovládáni.

Zbytek práce se v první části bude zabývat současným využitím programových logických automatů. Druhá část se bude zabývat programovatelnými logickými automaty jejich vstupy, výstupy a řídící logikou. Třetí část se zaměří na software sloužící k programování programovatelných logických automatů především na vývojový nástroj od německé firmy BECKHOFF TwinCAT. V této kapitole se také budeme zabývat programovacími jazyky, které jsou dané normou IEC/EN 61131-3. Čtvrtá část se zabývá návodem jak začít s TwinCATem pracovat a popisem vývojové části nástroje TwinCAT jménem PLC Control. Pátá část je pak celá zaměřená na popis vytvořeného programu Vrtačka a jeho následnou simulací. Šestá část se pak zabývá vytvořením a popisem programu robotická ruka v TwinCATu verzi 3 a po té pro tento program bude vytvořeno grafické rozhraní v jazyce C#. Výsledná simulace bude ovládána přes toto rozhraní a její výsledky budou vykresleny do grafů pomocí Scope Viewu od firmy Beckhoff[18].

1 SOUČASNÉ VYUŽITÍ PLC

Tato kapitola pojednává o současném využití PLC. V dnešní době může být pomocí PLC ovládána teplota v miniaturním automatizovaném zařízení při použití proporcionálního integračního derivačního (PID) kontroléru. Požadovaná teplota nebo nastavená hodnota se nastaví uživatelem pomocí HMI (human machine interface) a kontrolér v PLC se bude snažit udržet teplotu na nastavené hodnotě[1]. Teplota a vlhkost se také může řídit pomocí programovatelného automatu který je použit k realizaci fuzzy řízenou klimatizaci. Zpětnovazební signál je přes PLC přenesen do počítače pro fuzzy řízení. Poté se řídící signál z počítače, přenese na PLC pro ovládání rychlosti krokového motoru který simuluje provoz kompresoru. Tento systém může změnit rychlost pro různé zatížení a kdykoliv změnit nastavení kompresoru. U tohoto systému se teplota a vlhkost poskytnutá klimatizačním systémem udržuje na příjemné úrovni a snižuje zatížení klimatizace k dosažení úspory energie[14].

PLC by mohlo být využito i v Micro hydroelektronickém systému jako elektronický regulátor zatížení kvůli tomu že dokáže pracovat ve velmi prašném a vlhkém prostředí. V praxi by monitoroval frekvence generátoru, dynamicky by upravoval napájení systému zvýšením nebo snížením stanoveného výkonu odporové topné zátěži[2].

Využití PLC by mohlo být i ve fotovoltaice přesněji v solárním sledovacím systému s dvojitou osou který slouží pro natáčení solárního panelu směrem ke slunci. Tímto způsobem by se zvýšil výkon solárního panelu. Vertikální pohyb by sloužil pro sledování slunce přes den a horizontální pohyb by se staral o sledování pohybu slunce během roku[3].

PLC také může řídit pásové dopravníky pomocí nich můžeme přesouvat a třídit zboží[6]. Toto využití je použito u dopravníku zavazadel na letišti, také může byt použito i u výroby brambůrků kde PLC slouží jako automatické ovládání přístroje který slouží nejen k přesunu brambor, ale také k loupání, krájení a smažení brambor[4].

Elektrická ponorná čerpadla mohou byt nainstalovaná v ropných vrtech aby vytvořila umělý vztlak, pro zvýšení zotavení z nádrže a zvýšení rychlosti výroby. Model prediktivního regulátoru pro ESP, založený na linearizovaných modelech je vyroben a navržen s použitím SEPTIC (Statoiluv in-house softwarový nástroj pro model prediktivní regulace) a implementován do PLC[9].

Přes PLC se můžou ovládat i výtahy, kvůli složitosti logiky která ovládá výtah se logika napřed namodeluje pomocí MFG (Mark Flow Graph). Poté se MFG model přeloží do programovacího jazyku SFC (Sekvenční funkční diagram) kterému rozumí PLC[12].

Pro školní účely může být použita speciálně upravená souprava PLC, PLC Trainer který je upraven tak aby šlo pozorovat připojení mezi vstupy a výstupy, samotné PLC a výstupní moduly[7]. Za účelem zlepšení praktických schopností studentů. Pro tyto účely můžeme také použit automatický přenosový přepínač (ATS), který je vytvořený na základě PLC[8]. ATS může být použito i jako samostatný systém v komerčních a průmyslových praktických oblastech.

PLC své místo dokáže najít i ve vozidlech, které se starají o přepravu materiálu v továrnách. Pro zlepšení automaticky naváděných vozidel bylo navrhnuto použití integrovaného logického fuzzy kontroléru s programovatelným logickým automatem (IFLPLC). Nejenom, že tímto způsobem se zlepšila flexibilita vozidla, bylo také docíleno velkých výhod, které se v budoucnu mohou využít. Automatické naváděná vozidla využívající PLC a FLC dohromady, mohou být velkým přínosem do továren využívajících pružné výrobní systémy.[10]

Pro zlepšení rozvodu vody se může použít PLC, které se stará o tlakový reaktor. Pracuje jako záloha operačního systému mezi PLC a analogově řídícím systémem, překonává nevýhody PLC modulu typu TI545 a realizuje pokročilý řídící algoritmus pro zlepšení ovládání výkonu. Implementace indikuje, že řídicí systém PLC zlepšuje výkon jak regulátoru vodní hladiny parních generátorů, tak i poruchy odmítnutí[11].

Řízení a sledování strojů na dálku je důležitým oborem v těchto dnech. Kvůli evoluci v komunikačních systémech se vědci pokouší tyto systémy implementovat do strojů z důvodů umožnění ovládání a monitorování zařízení pomocí mobilní telefonní sítě. PLC by bylo ovládáno pomocí GSM dat ve formě SMS zprávy, přes tuto zprávu by bylo možné zařízení vypnout, zapnout a nebo zjistit stav zařízení[13].

Existují vzdálené laboratoře pro systémové inženýrství a kontrolní kurzy automatizace, založené na kombinovaném použití TwinCATu, laboratorního Java serveru a jednoduché Java simulace (EJS). TwinCAT systém se používá pro uzavření regulační smyčky pomocí programovatelných automatů rozmístěných v počítačích, které mají TwinCAT run-time nástroj. EJS se používá k rozvoji laboratorních front-end appletů, které umožňují učitelům a studentům parametrizovat a sledovat chování PLC z libovolného počitače. Laboratorní java servery zajišťují spojení mezi applety EJS a PLC, současně zajišťuje individuální přístup ke každému PLC[15].

2 PLC

PLC (Programmable Logic Controller) viz obrázek 2.1 je číslicový elektronický systém využíván v průmyslovém odvětví, uzpůsobený pro potřeby řešení automatizačních úloh v reálném čase, s co nejkratší dobou odezvy. Používá se k automatizaci a komunikaci s dalším zařízením k němu připojením. Ke komunikaci s okolím má PLC vstupní porty na které jsou přivedeny signály z řízeného procesu, přivedené signály mohou být binární v podobě stavu zapnuto/vypnuto (příklad binárního použití může být snímání polohy koncovým snímačem) nebo v podobě spojitých analogových signálů (například teplota, tlak, hladina a podobné).

Po té má PLC výstupní porty ke kterým jsou připojeny akční prvky řízeného procesu a stejně jako u vstupních portů mohou být na nich řídící signály binární zapnuto/vypnuto (stykač motoru, cívka ventilu ...) nebo spojitě analogové (řízení rychlosti, polohy regulačního ventilu ...). Mezi vstupy a výstupy je řídící logika (CPU), která na základě vstupů ovládá výstupy. Aby PLC reagovalo na stav vstupních signálu musí do jeho pamětí být uložen algoritmus (program) který vytvoří programátor pomocí programovacího softwaru.

Obr. 2.1: BC9191 pokojový regulátor. Převzato z [16].

2.1 Nejznámější výrobci PLC:

- Česká společnost:
  - TECO[1]
- Německé firmy:
  - BECKHOFF[2]
  - SIEMENS[3]
- Japonské společnosti:
  - Omron[4]
  - Mitsubishi[5]
- Americká společnost:
  - Rockwell automation/Allen Bradley[6]

PLC si můžeme koupit už na programované anebo si je můžeme sami naprogramovat, k tomu slouží programovací prostředí: TwinCAT od BECKHOFF, Step7 od Siemens, SG2 od TECO, CX-programmer od Omron, GX IEC Developer od Mitsubishi a RSLogix od Rockwell automation.

[1]Domovská stránka na URL: http://tecomat.com
[2]Domovská stránka na URL: http://beckhoff.com
[3]Domovská stránka na URL: http://siemens.com
[4]Domovská stránka na URL: http://industrial.omron.eu
[5]Domovská stránka na URL: http://eu3a.mitsubishielectric.com
[6]Domovská stránka na URL: http://ab.rockwellautomation.com

3 TWINCAT

TwinCAT (The Windows and Control Automation Technology) je vývojový nástroj a run-time pro PLC a IPC od německé společnosti BECKHOFF který pracuje v rámci normy IEC 61131-3 ve které jsou doporučovány čtyři programovací jazyky s přesně definovanou sématikou a syntaxí LD, FBD, IL a ST. Jako pátý programovací jazyk se často uvádí SFC. Někteří výrobci nabízejí kromě těchto jazyků i možnost programování v dalších jazycích, které nejsou v normě obsaženy, ale uživatelům mohou přinést určité výhody (např. S7-Graph, CFC).

V dnešní době jsou na trhu dvě verze TwinCATu, verze 2 (na kterou se zaměříme) a verze 3. TwinCAT běží jako služba na pozadí systému Windows. Má dvě části, konfigurační část programu (System manager) a prostředí pro samotný návrh programu (PLC control).

System manager má za úkol nastavení komunikace mezi počítačem a PLC (IPC). Můžeme zde nastavovat průmyslové sběrnice na zařízení, vstupy, výstupy a parametry všech přídavných karet.

PLC kontrolér slouží k vytváření samotného programu. Umožňuje přenos programu do a z IPC, jeho spuštění a především jeho simulací. Při simulaci programu je možné za běhu měnit hodnoty proměnných a sledovat odezvu programu. Pomocí tohoto prostředí můžeme také tvořit jednoduché vizualizace, které budou upozorňovat na jednotlivé stavy a funkce řídícího programu.

3.1 Přehled programovacích jazyků daných normou IEC/EN 61131-3:

3.1.1 Jazyk seznamu instrukcí IL-Instruction List

Patří do skupiny textových jazyků. Také bývá označován jako jazyk pokynů (povelů), neboli seznam instrukcí. Programová organizační jednotka je složena ze sekvence instrukcí, z nichž každá začíná na novém řádku, může obsahovat také komentář. Pomocí modifikátorů se vyjadřují negace, podmíněnost a nepodmíněnost instrukce skoků, volání a priorita [17].

3.1.2 Jazyk strukturovaného textu ST-Structured Text

Je vyšší programovací jazyk, který má kořeny v jazycích Pascal a C. Syntaxe jazyka je dána povolenými příkazy a výrazy ukázka v tabulce 3.2. Příkazy jsou odděleny středníkem a může jich být více na jednom řádku. ST jazyk je vhodným nástrojem

pro definování komplexních funkčních bloků, které pak mohou být použity v libovolném programovacím jazyku [17].

Tab. 3.1: Seznam příkazů jazyka strukturovaného textu ST

Příkaz	Popis	Příklad	Poznámka
`:=`	Přiřazení	A:=22;	Přiřazení hodnoty vypočtené na pravé straně do identifikátoru na levé straně
	Volání funkčního bloku	InstanceFB (par1:=10, par2:=20);	Volání funkčního bloku s předáváním parametrů
`IF`	Příkaz výběru	IF A > 0 THEN B:=100; ELSE B:=0; END_IF;	Výběr alternativy v podmíněný výrazem BOOL
`CASE`	Příkaz výběru	CASE kod OF 1 : A:=11; 2 : A:=22; ELSE A:=99;	Výběr bloku příkazů podmíněný hodnotou výrazu „kod“
`FOR`	Iterační příkaz smyčka FOR	FOR i:=0 TO 10 BY 2 DO j:=j+i; END_FOR;	Vícenásobná smyčka bloku příkazů s počáteční a koncovou podmínkou a hodnotou inkrementu

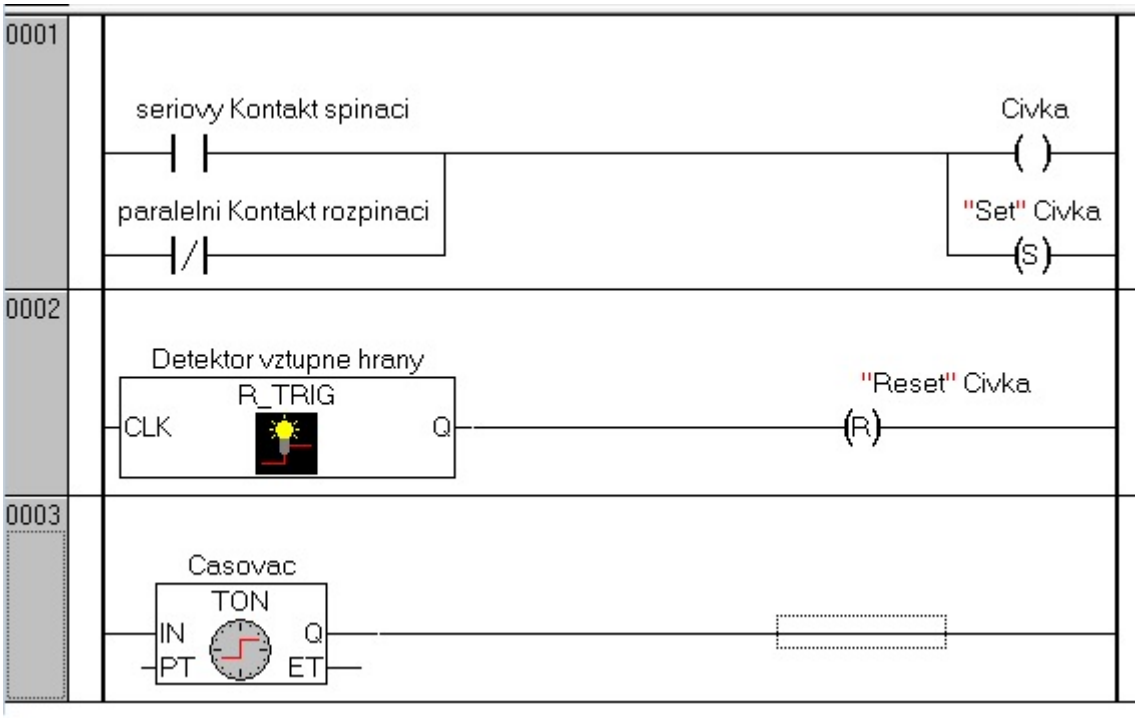
`WHILE`	Iterační příkaz smyčka WHILE	WHILE i>0 DO
		n:=n*2;
		END_WHILE;
			Vícenásobná smyčka bloku příkazů s podmínkou ukončení smyčky na začátku

`REPEAT`	Iterační příkaz smyčka REPEAT	REPEAT
		k:=k+i;
		UNTIL i<20;
		END_REPEAT;
			Vícenásobná smyčka bloku příkazů s podmínkou ukončení smyčky na konci

`EXIT`	Ukončení smyčky	EXIT;	Předčasné ukončení iteračního příkazu
`RETURN`	Návrat	RETURN;	Opuštění právě vykonávané POU a návrat do volající POU (programové organizační jednotky)
`;`	Prázdný příkaz	;;	

3.1.3 Jazyk příčkového diagramu LD-Ladder Diagram

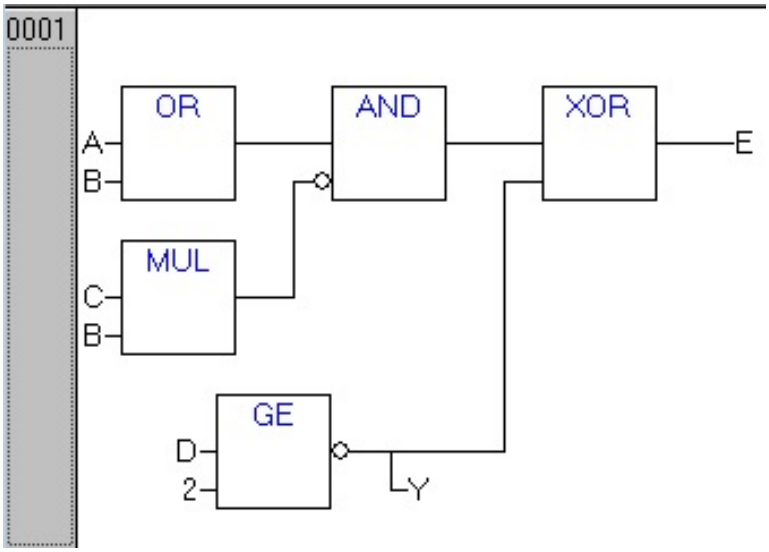
Také nazývaný jako jazyk kontaktních schémat, je založen na grafické reprezentaci reléové logiky [17]. Program je napsán pomocí grafických prvků propojených v sítí. Tato síť je zleva i zprava ohraničena svislými čarami, které se nazývají napájecí sběrnice. Mezi nimi je takzvaná příčka, která může být rozvětvena. Každý úsek příčky, vodorovný nebo svislý, je ve stavu „zapnuto" nebo „vypnuto". Do příček mohou být vloženy kontakty (spínací, rozpínací apod.), cívky (cívka, negovaná cívka apod.), funkce a funkční bloky viz obrázek 3.1.



Obr. 3.1: Ukázka programu v jazyce LD v TwinCatu

3.1.4 Jazyk funkčního blokového schématu FBD-Function block diagram

Vyjadřuje chování funkcí, funkčních bloků a programů jako soubor vzájemně provázaných grafických bloků podobně jako v elektronických obvodových diagramech [17]. Používají se zde standardní funkční bloky pro vyjádření logických funkcí a také čítače, časovače, komunikační bloky a podle potřeby i speciální bloky. Dostupné bloky se mohou lišit podle výrobce viz obrázek 3.2



Obr. 3.2: Ukázka programu v jazyce FBD v TwinCATu

3.1.5 Jazyk volně propojovaných bloků CFC-Continuous Function Chart Editor

Podobný grafický jazyk na vysoké úrovni jako FBD, ale navíc dokáže vytvářet vícevláknové algoritmy, které lze počítat paralelně z hlediska programátora. V CFC editoru mohou být na pracovní plochu vloženy, zkonfigurovány a za podpory funkce autorouting vzájemně propojeny předpřipravené funkční bloky [17].

3.1.6 Sekvenční funkční diagram SFC-Sequential Function Chart

Popisuje sekvenční chování řídicího programu. Odvozen ze symboliky Petriho sítí. (Matematická reprezentace diskrétních distribuovaných systémů. Graficky reprezentuje strukturu distribuovaného systému jako orientovaný bipartitní graf s ohodnocením. Má dva druhy uzlů označované jako místa a přechody a orientované hrany

spojující místa s přechody[5].) SFC umožňuje rozložit úlohu řízení na zvládnutelné části a zachovat přitom přehled o chování celku .

Skládá se:

- Krok:
  - Reprezentuje stav řízeného systému a má k sobě přiřazen blok akcí.
- Přechod:
  - Je spojen s podmínkami, které musí být splněny, aby mohl být deaktivován krok, který přechodu předchází, a naopak aktivován krok, který následuje.

Každý prvek může být naprogramován v libovolném jazyku definovaném v normě, včetně vlastního SFC. Jazyk umožňuje i větvení programu se spojením alternativních větví a paralelní souběh více větví s jejich následnou synchronizací [17].

3.2 Komunikace:

Propojení přes sběrnice:

- EtherCAT
- PROFIBUS DP/MC
- CANopen
- DeviceNet
- INTERBUS
- SERCOS
- Lightbus
- Ethernet
- PC hardware (paralelní port, sériový port, USB)

3.3 Systémové informace o TwinCATu 2:

Tab. 3.2: Přehled produktu TwinCAT: Software PLC

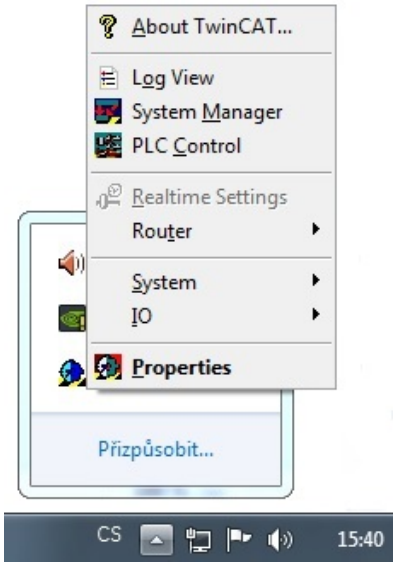
	TwinCAT PLC
PC hardware	standard PC/IPC hardware, žádné doplňky
Real-time	Beckhoff real-time kernel
Vstupní výstupní systém	EtherCAT, Lightbus, PROFIBUS DP/MC, Interbus, CANopen, DeviceNet, SERCOS, Ethernet
Run-time systém	4 víceúlohové PLC každé se 4 úkoly v každém PLC run-time systému, vývoj a run-time systémy na jednom PC nebo odděleně
Paměť	Velikost obrazu procesu, oblast příznaků, velikost programu, velikost organizačního programu jednotky, počet proměnných omezen pouze velikostí uživatelské paměti maximálně 2 GB pro NT/2000/XP/Vista)
Doba cyklu	nastavitelná od 50 μs
Link-time	1 μs (Intel® Core™2 Duo)
Programování	IEC 61131-3: IL, FBD, LD, SFC, ST, výkonné řízení knihoven, pohodlné ladění

4 SIMULACE V TWINCAT 2

TwinCAT 2 se skládá ze dvou částí:

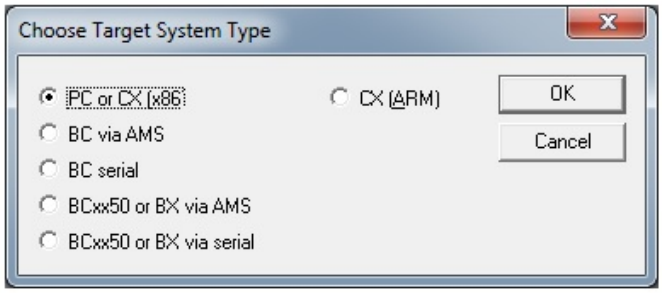
- Run-time - real-time řídící systém
- Vývojové prostředí pro tvorbu a konfiguraci projektu

Projekty vytvořené v TwinCATu se skládají ze dvou částí, softwarové a hardwarové. Softwarová se vytváří v PLC Control a je uložena v souboru s příponou *.pro*. Hardwarová část se vytváří v Systém Manageru a je uložena v souboru s příponou *.tsm*. Obě tyto aplikace se spustí z hlavního panelu mezi ikonami oznamovací oblasti, ukázka na obrázku A.1.



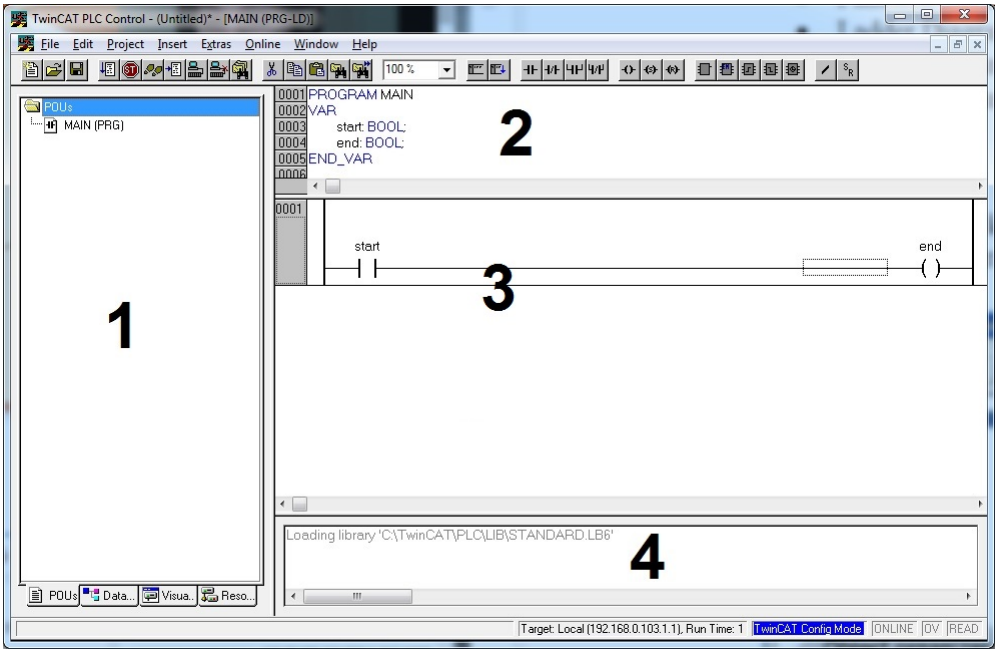
Obr. 4.1: Ukázka spuštění TwinCATu přes hlavní panel

Při spuštění je potřeba vybrat hardwarovou platformu (PC, BC, CX) Obr.4.2.



Obr. 4.2: Vybrání hardwarové platformy

Po vybrání platformy se otevře samotný PLC control, kde se bude tvořit PLC program.



Obr. 4.3: TwinCAT PLC Control s jednoduchým programem v LD

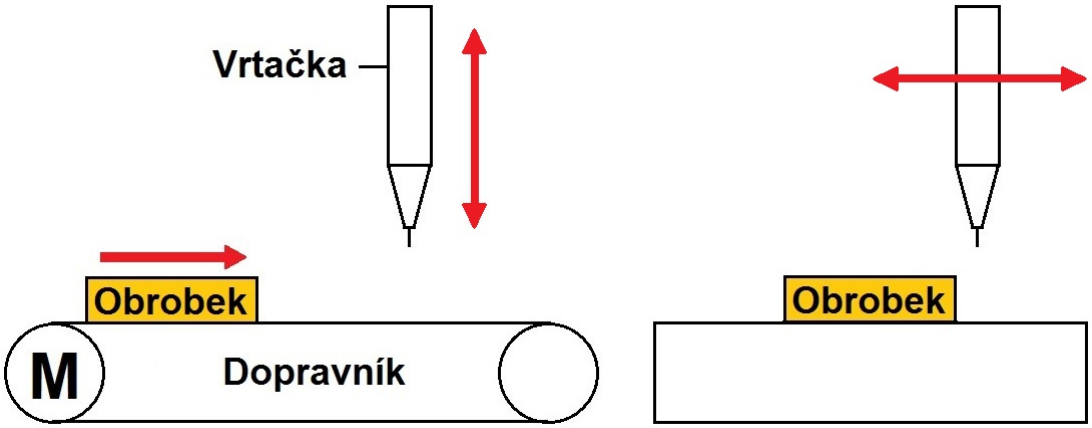
4.1 Rozdělení programu PLC control:

1. Objekt organizer se čtyřmi záložkami
   - Záložka POUs: Obsahuje všechny uživatelské programové jednotky(programy, funkční bloky, funkce, akce).
   - Záložka Data types: Obsahuje uživatelské typy.
   - Záložka Visualisations: Obsahuje pomocné vizualizace.
   - Záložka Resources: Obsahuje globální proměnné, seznam použitých knihoven, konfigurace alarmu, konfigurace úloh a pomocné funkce na monitorování proměnných.
2. Slouží na deklarování lokálních proměnných, stavy proměnných.
3. Síť(network) která obsahuje řídící funkce, tvořící samotný PLC program.
4. Slouží na zobrazení informací pro programátora(upozornění, chyby).

Po dokončení všech potřebných částí programu se musí zaškrtnout Online-Simulation mode, tím se program přepne do simulačního módu kde zmizí potřeba mít reálné PLC. Po té se vytvořený projekt zapíše do PLC volbou Online-login a spustí se pomocí volby Online-run nebo klávesou F5.

5 VYTVOŘENÍ SIMULACE V TWINCATU 2

Tato kapitola pojednává o vytvoření simulace programu 5.1 který má základ v již vytvořeném programu od BECKHOFFu sample machine, který se vytvoří při instalaci TwinCATu 2.11. Dále zde budou popsány a vysvětleny jednotlivé bloky vytvořeného programu. Po té zde bude popsána simulace tohoto programu a na konci kapitoly bude graf popisující práci simulace.



Obr. 5.1: Náčrt modelu Vrtačky

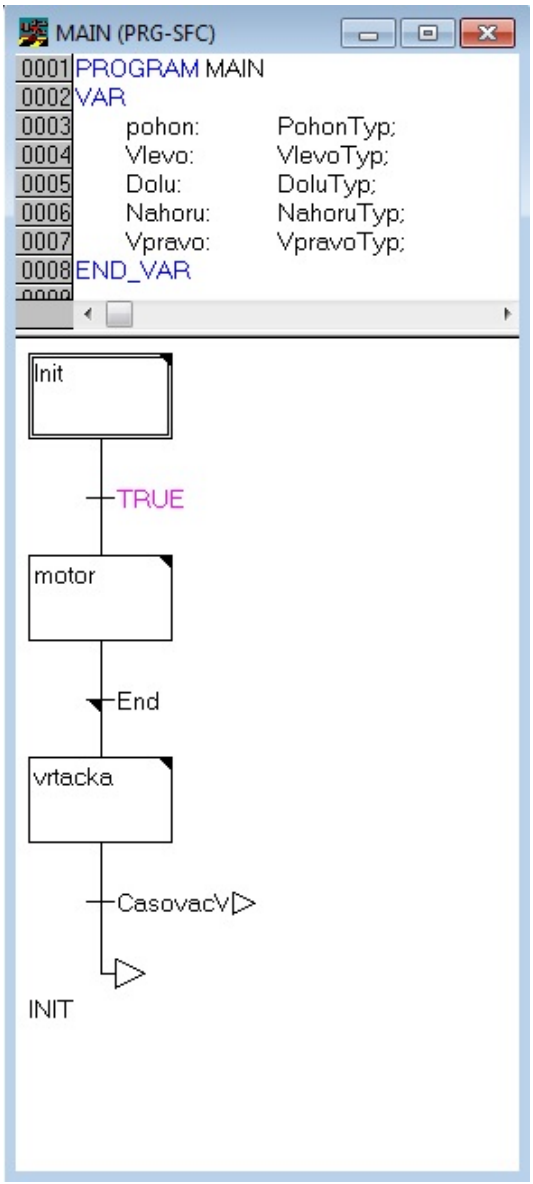
Popis:

1. Dopravní pás se pohne o 20 kroků.
2. Vrtačka se posune doleva.
3. Vrtačka se posune dolů v intervalu 2 sekund.
4. Vrtačka se posune zpět nahoru.
5. Vrtačka se posune doprava.
6. Vše začne znovu od kroku jedna.

Úpravou programu se přidaly body 2 a 5 které posouvají vrtačku horizontálním směrem. Tato úprava vznikla z důvodu tvaru obrobku u kterého by nestačil jen vertikální posun.

5.1 Vytvořený program

Program začíná vytvořením programové jednotky MAIN, která slouží jako hlavní část programu viz obrázek 5.2. Tato jednotka se skládá ze 3 bloků (Init, motor, vrtačka) a na konci je skok který skáče zpátky na začátek programu na Init. CasovacVpravo.Q je standardní výstup časovače.



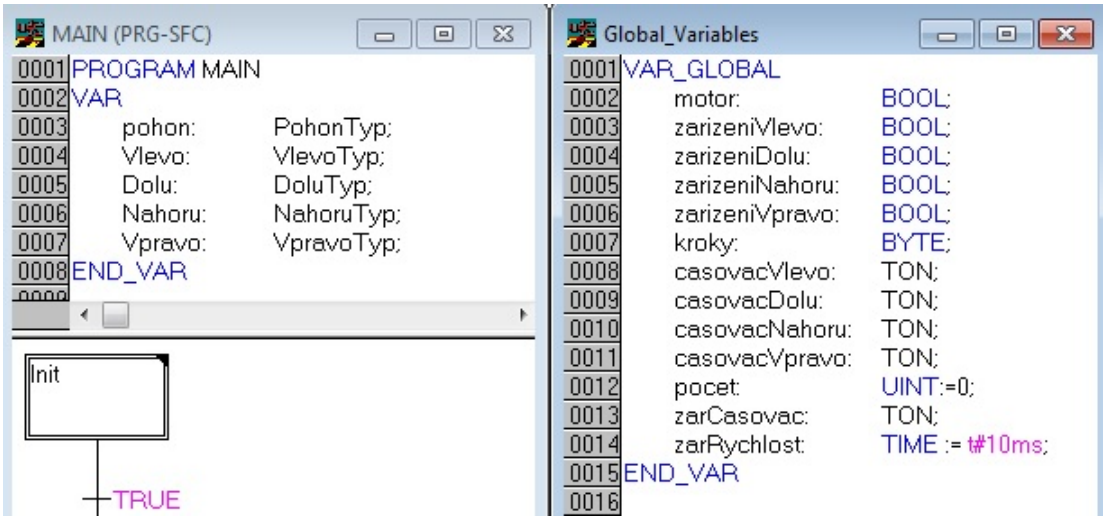
Obr. 5.2: Programová jednotka MAIN

5.1.1 Programové proměnné

Proměnné mohou být přiřazeny explicitně k hardwarovým adresám (vstupům, výstupům) pouze v konfiguracích, zdrojích nebo programech. Tímto způsobem je dosaženo vysokého stupně hardwarové nezávislosti a možnosti opakovaného využití softwaru na různých platformách.

Oblast působnosti proměnných je běžně omezena pouze na tu programovou organizační jednotku, ve které byly deklarovány (lokální proměnné). To znamená, že jejich jména mohou být používána v jiných částech bez omezení. Pokud mají mít proměnné globální působnost, musí být deklarovány jako globální (VAR_GLOBAL). Aby bylo možné správně nastavit počáteční stav procesu nebo stroje, může být parametrům přiřazena počáteční hodnota při startu nebo restartu.

Na obrázku 5.3 jde vidět přehled proměnných a jejich deklarace, v levé části obrázku jsou lokální proměnné a v pravé části jsou globální proměnné.



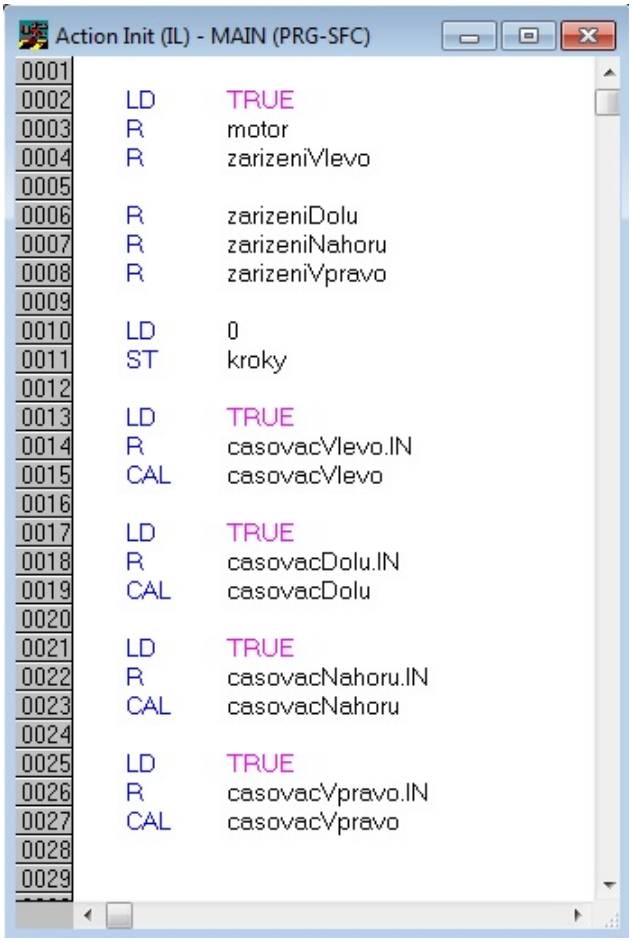
Obr. 5.3: Přehled proměnných a jejich deklarace

Popis deklarace:

- pohon – Deklarován jako programová jednotka PohonTyp.
- motor – Deklarován jako datový typ BOOL.
- kroky – Deklarován jako datový typ BYTE.
- casovacVlevo – Deklarován jako funkční blok TON(Časovač).
- pocet – Deklarován jako celočíselná hodnota 0.
- zarRachlost – Deklarován jako datový typ TIME s hodnotou 10 ms.

5.1.2 Inicializační část

První programovatelný blok který inicializuje proměnné, je nastaven jako IL. Vždy když program projde tímto blokem tak se proměnné resetují do nastavených hodnot, tímto se zamezí chybným změnám hodnot.



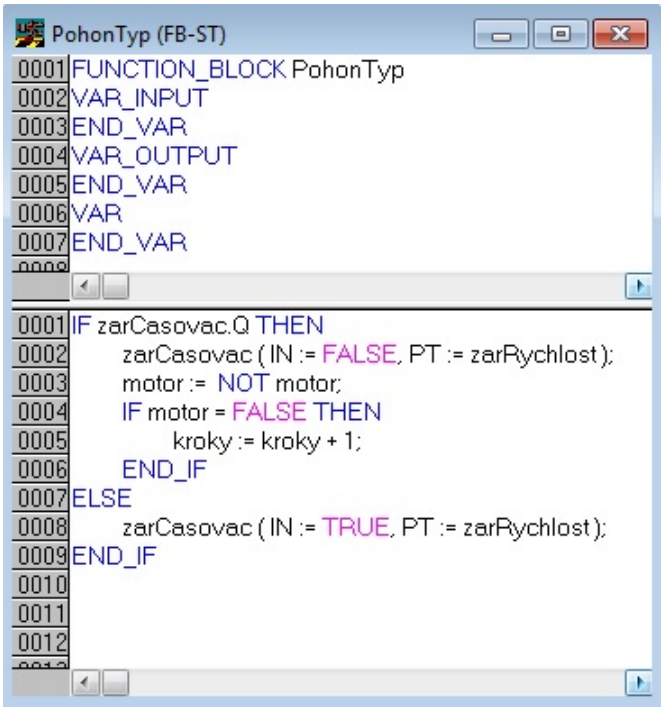
Obr. 5.4: Seznam instrukcí pro Init

Popis použitých operátorů:

- LD – Nastaví aktuální výsledek na hodnotu rovnou operandu.
- R – Smaže boolovský operand na „0“ Operace se provede pouze pokud je aktuální výsledek boolovská „1“.
- ST – Uloží aktuální výsledek na místo operandu.
- .IN – Vstup časovače.
- CAL – Volání funkčního bloku.

5.1.3 Kód programového bloku motor

Programový blok který se stará o dopravní pás, blok je nastaven jako ST. V sobě má napsán příkaz: pohon(), pomocí kterého se spustí programová jednotka PohonTyp.

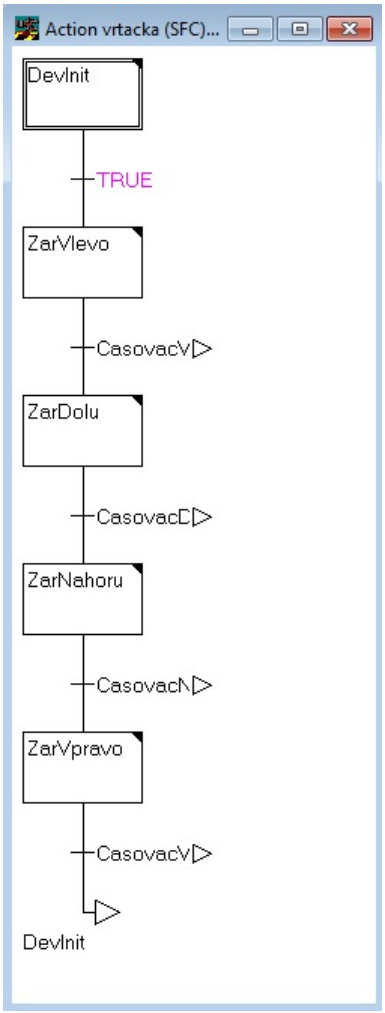


Obr. 5.5: Program v programové jednotce PohonTyp

End přechod který je za motorem určuje zda již bylo dosaženo 20 kroků motoru. Jestli tohoto bylo dosaženo program pokročí a přejde v novém cyklu z motoru na vrtání.

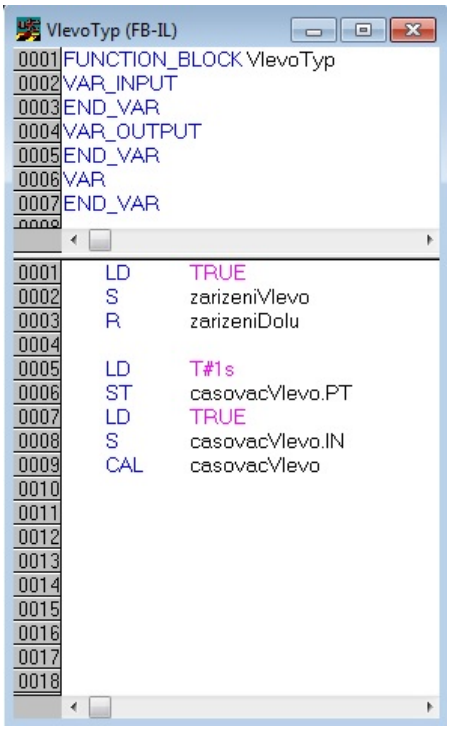
5.1.4 Kód programového bloku vrtačka

Programový blok, který se stará o posuny vrtačky. Je nastaven jako SFC a skládá se z více programových bloků, každý blok se stará o jeden pohyb vrtačky.



Obr. 5.6: Vnitřek programového bloku vrtačka

DevInit vypne blok motor, *motor* = *FALSE* a bloky *ZarVlevo*, *ZarDolu*, ZarNahoru a *ZarVpravo* volají programové jednotky *VlevoTyp*, *DoluTyp*, NahoruTyp a *VpravoTyp* pomocí operátoru CAL 5.7. *CasovacVlevo.Q*, *CasovacDolu.Q*, *CasovacNahoru.Q*, *CasovacVpravo.Q* jsou standardními výstupy časovače.



Obr. 5.7: Vnitřek programové jednotky VlevoTyp

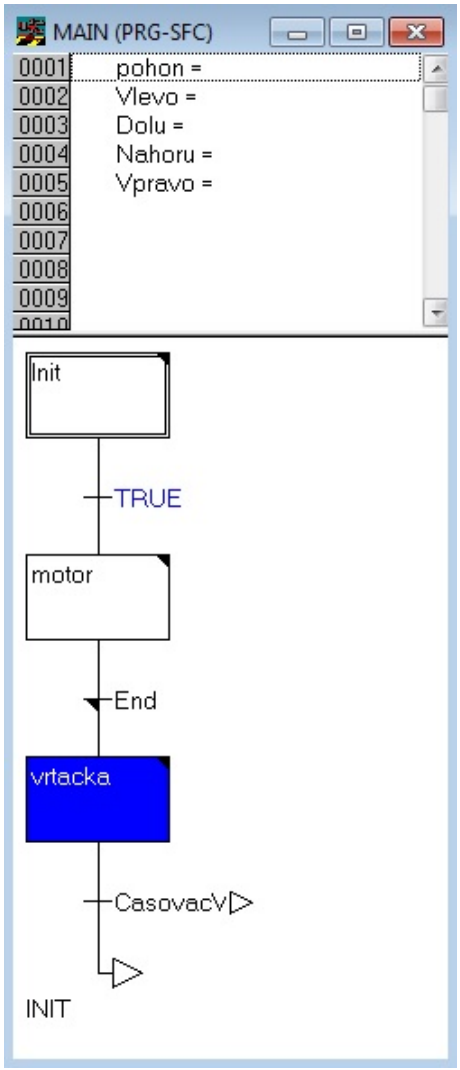
Tyto jednotky jsou nastavené jako IL, jsou si sobě velmi podobny jen s malým rozdílem v proměnných a časem trvání bloku.

Popis použitých operátorů:

- LD – Nastaví aktuální výsledek na hodnotu rovnou operandu.
- S – Nastaví boolovský operand na „1“ Operace se provede pouze pokud je aktuální výsledek boolovská „1“.
- R – Smaže boolovský operand na „0“ Operace se provede pouze pokud je aktuální výsledek boolovská „1“.
- ST – Uloží aktuální výsledek na místo operandu.
- CAL – Volání funkčního bloku.
- .PT – Předvolba časovače.
- .IN – Vstup časovače.

5.2 Vytvořená simulace

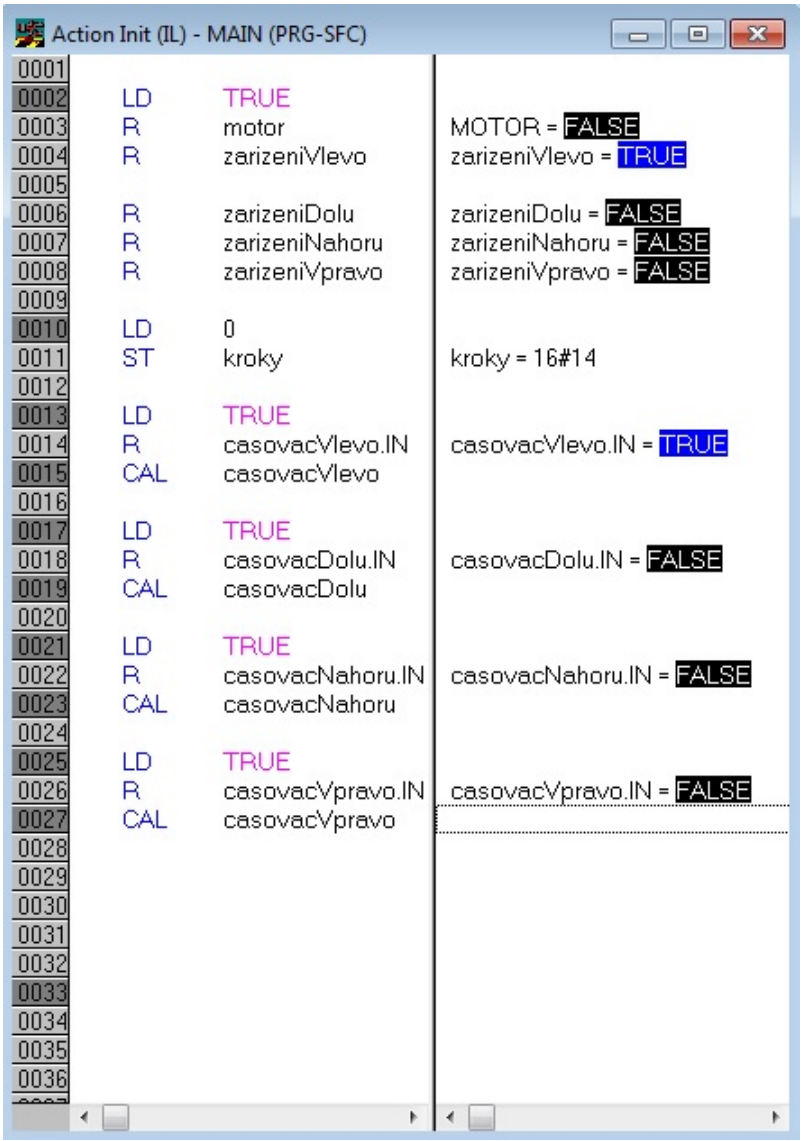
Simulace se zapíše do virtuálního PLC pomocí volby Online-Login, po té se projekt spustí volbou Online-Run. Spuštěním programu se spustí i simulace která ukazuje ve které části program zrovna pracuje.



Obr. 5.8: Simulovaný program probíhající v programové jednotce Main

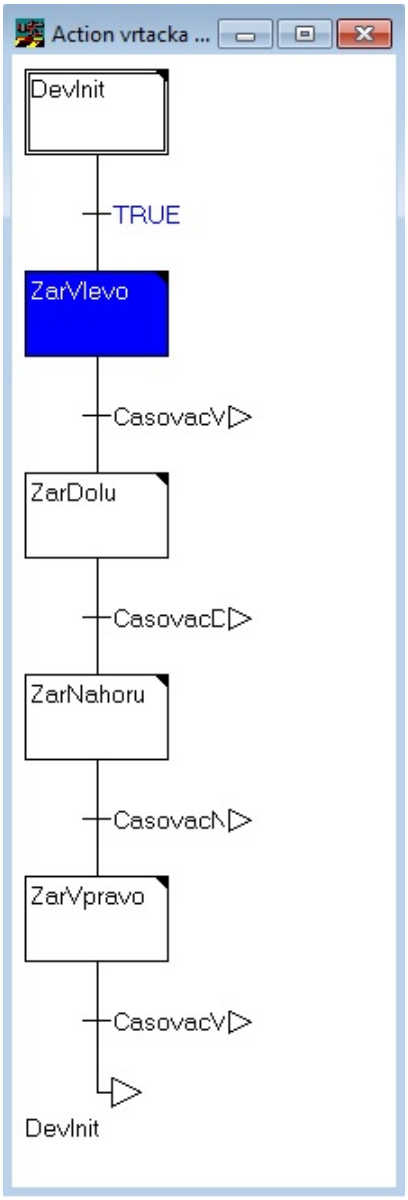
Právě vykonávaná část simulace je označena modrou barvou. Z obrázku 5.8 lze zjistit, že pracující blok je blok vrtačka, bloky Init a motor již byly vykonány a jejich hodnoty upraveny podle programu.

Z obrázku 5.9 jdou vidět změny v hodnotách proměnných, Proměnné které zrovna pracují jsou označené hodnotou TRUE.



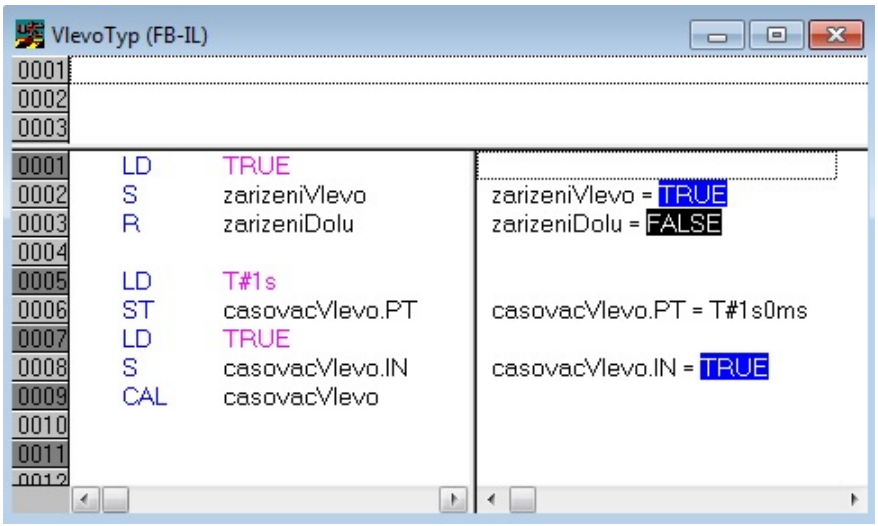
Obr. 5.9: Upravené hodnoty v bloku Init

V bloku vrtačka právě pracuje blok ZarVlevo, který se stará o posunutí vrtačky o sekundu doleva. Blok DevInit nastavil proměnnou motor na FALSE, tím to vypnul motor a zastavil se dopravní pás.



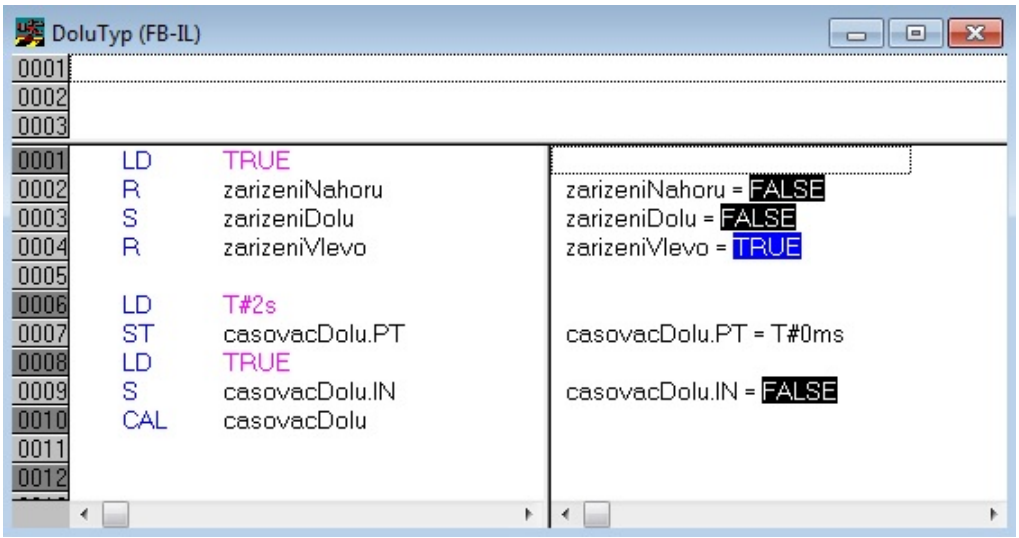
Obr. 5.10: Simulovaný program probíhající v bloku Vrtačka

Přes funkci CAL začal pracovat funkční blok VlevoTyp, který nastavil proměnné zarizeniVlevo, casovacVlevo.IN na hodnotu TRUE a zarizeniDolu nastavil na FALSE. Do předvolby časovače casovacVlevo nastavil hodnotu 1 sekunda a po té ho zavolal.



Obr. 5.11: Simulovaný program v jednotce VlevoTyp

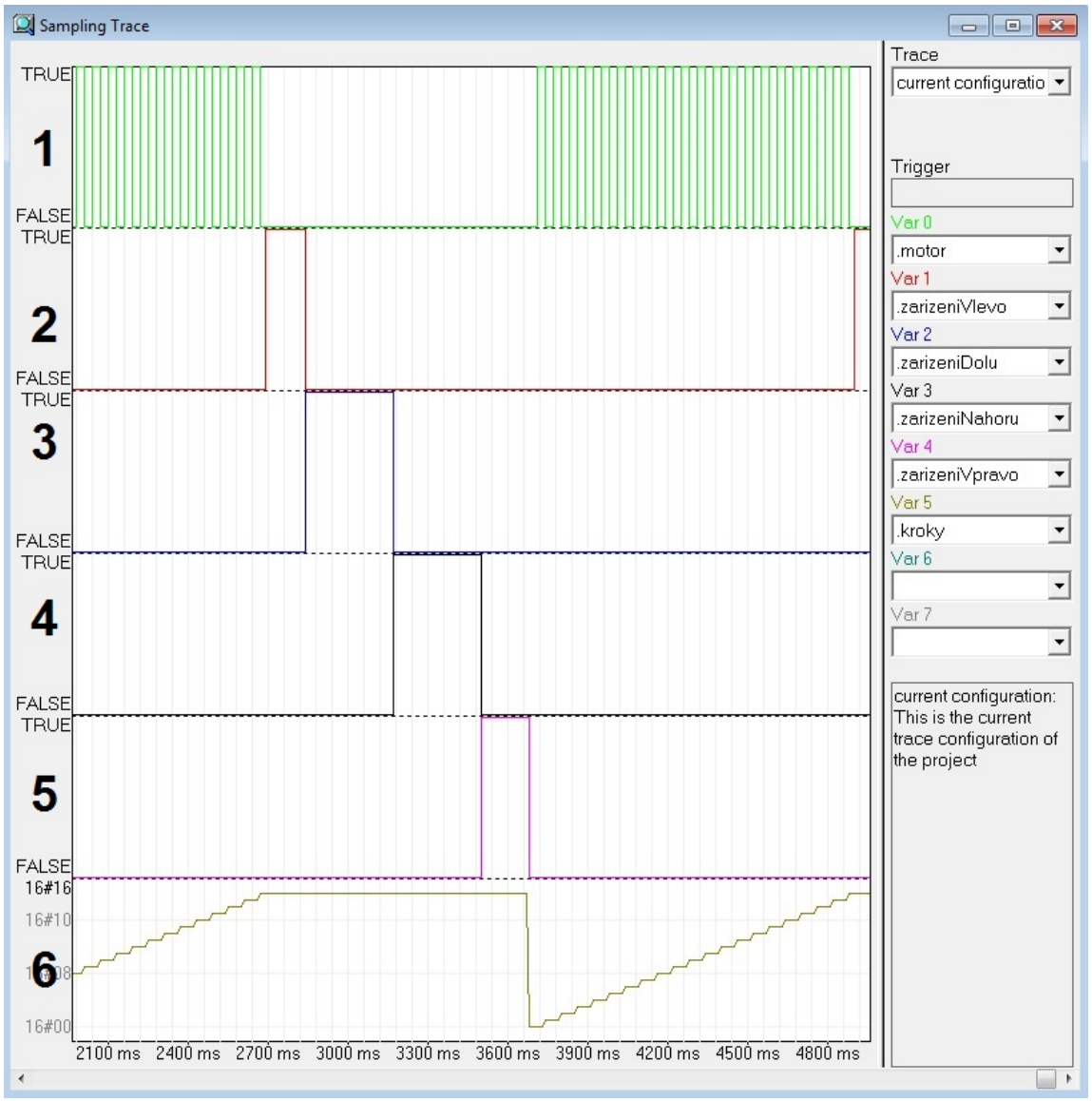
Ostatní bloky čekají na dokončení předešlého bloku. Blok ZarDolu který je pod blokem ZarVlevo je v v klidovém stavu. ZarDolu pomocí příkazu CAL volá jednotku DoluTyp.



Obr. 5.12: Klidový stav jednotky DoleTyp

Po provedení posledního bloku se program vrací zpět na začátek, odkud začne opět pracovat.

5.2.1 Graf simulace

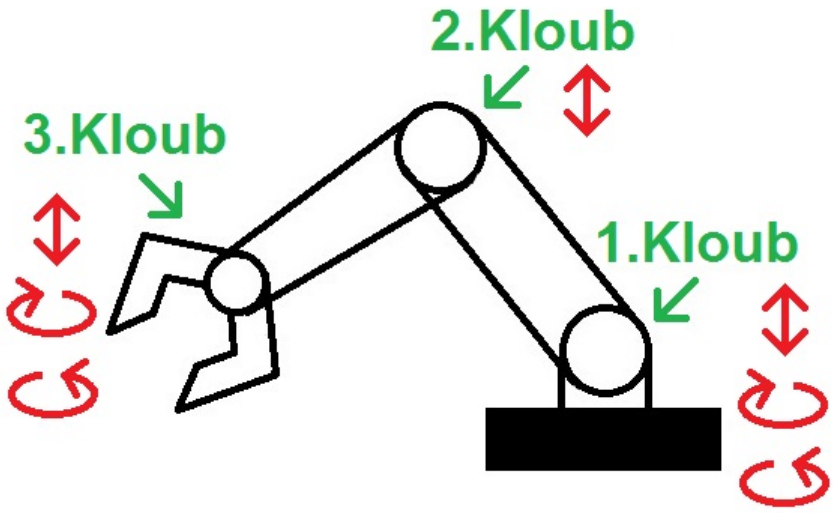


Obr. 5.13: Grafy simulace z programu TwinCAT

Popis grafu:

1. Graf motoru – Motor jede do doby než se dosáhne 20 kroků.
2. Graf zarizeniVlevo – Doba posunutí vrtačky doleva.
3. Graf zarizeniDolu – Doba posunu dolů včetně vrtání.
4. Graf zarizeniNahoru – Doba posunu nahoru.
5. Graf zarizeniVpravo – Doba posunu doprava (na původní místo).
6. Graf kroku – Graf počítadla kroků. Maximum kroků je 20.

6 VYTVOŘENÍ PROGRAMU A GRAFICKÉHO ROZHRANÍ PRO OVLÁDÁNÍ ROBOTICKÉ RUKY

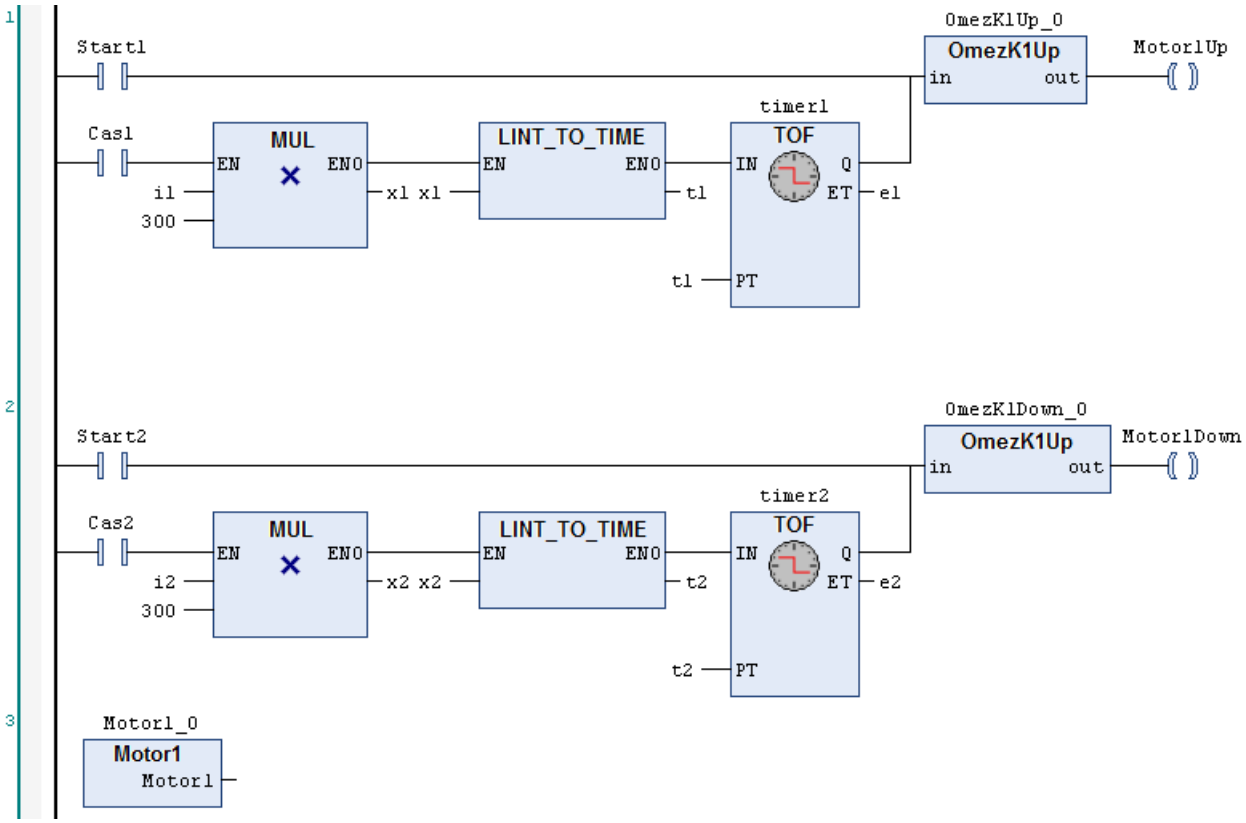


Obr. 6.1: Robotická ruka

Simulovaná robotická ruka je tvořena z 6 motorů. První kloub (rameno) má 2 motory, jeden pro ovládání horizontálního točení doprava nebo doleva. Druhý motor slouží pro vertikální pohyb směrem nahoru nebo dolů. Druhý kloub má jen jeden motor který otáčí ruku vertikálním směrem nahoru nebo dolů. Třetí kloub má 3 motory, dva z nich slouží pro pohyb ruky horizontálním a vertikálním směrem a poslední motor slouží pro zavíraní nebo otevírání ruky. Tyto motory bude řídit PLC, které bude naprogramováno v programovém prostředí TwinCAT. Tento program bude v další části popsán a vysvětlen, konec kapitoly se zaměří na grafické rozhraní pomocí které ho se řídí program v TwinCATu. Podkapitola grafického rozhraní bude rozdělena na výběrové menu, které slouží pro výběr ovládání (manuální a automatické) a poté tyto typy ovládání budou jednotlivě popsány.

6.1 Vytvořený program v prostředí TwinCAT

Hlavní část programu tvoří programová organizační jednotka *Main*, která je vytvořena jako příčkový diagram (Ladder diagram). V této jednotce jsou naprogramovány všechny pohyblivé části robotické ruky. Na obrázku 6.2 je ukázána část jednotky *Main* která se stará o vertikální pohyb prvního kloubu.



Obr. 6.2: Část programové organizační jednotky: Main

Popis jednotlivých částí:

- Start1/2 – Kontakt sloužící jako tlačítko v manuálním módu pro zapnutí Motor1Up/Down přes objekt OmezK1Up/Down.
- Cas1/2 – Kontakt sloužící jako tlačítko v automatickém módu který slouží pro zapnutí časovače Timer1/2.
- i1/2 – Proměnné do kterých se nahrají hodnoty poslané z grafického rozhraní, dále tyto hodnoty putují přes bloky Mul a Lint to Time do časovače timer, kde se stanou proměnou t1/2. Do které proměnné (i1 nebo i2) se zapíše určuje zadané znaménko v grafickém rozhraní.
- Mul – Blok starající se o zvýšení vstupní hodnoty, tato hodnota je zvětšena z důvodu navýšení doby po kterou je časovač Timer1/2 otevřen.

- Lint to Time – Blok starající se o převod hodnoty z datového typu integer na čas.
- Timer1/2 – Časovač typu Tof (Timer Off-Delay), tento časovač začne pracovat(časovat) v okamžiku spádové hrany řídící proměnné IN do doby t1/2.
- Motor1Up/Down – Výstupní kontakt pro řízení motoru.

6.1.1 Bloky omezující pohyb robotické ruky

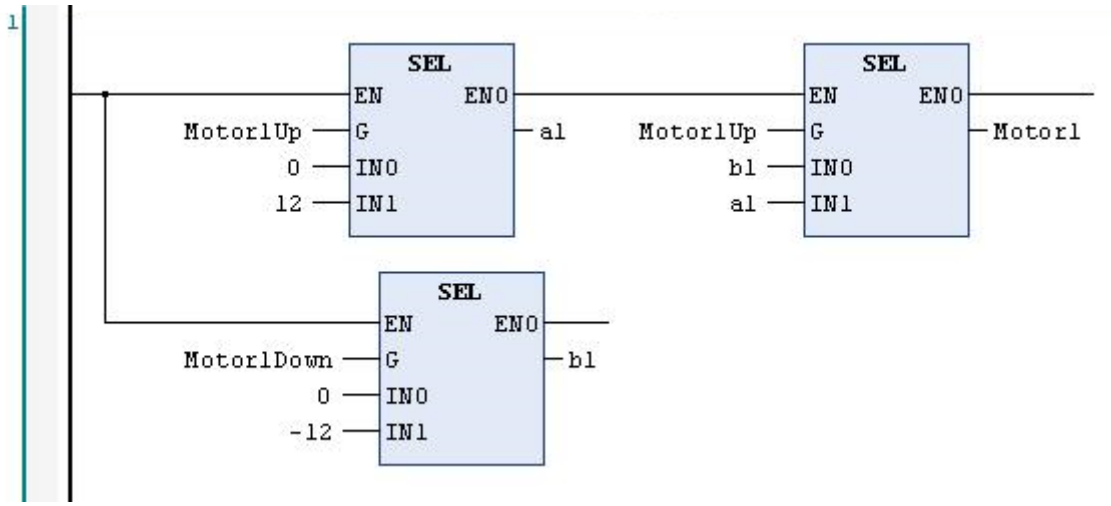
Kód programového bloku Omezk1Up:

```
IF in=TRUE
THEN
zk1Ver:=zk1Ver+1;
     IF zk1Ver<=5000
          THEN
               out:=TRUE;
          ELSE
               out:=FALSE;
               zk1Ver:=5030;
               t1:=Time#0MS;
          END_IF
ELSE
     out:=FALSE;
END_IF
```

Programový blok starající se o omezení pohybu ruky. Na začátku podkapitoly jde vidět kód programového bloku *OmezK1Up*, který se stará o omezení ruky při pohybu nahoru. Při aktivaci tohoto bloku se spustí počítadlo *zk1Ver* a při splnění podmínky se nastaví privátní proměnná *Out* na hodnotu *TRUE*.

Když proměnná *zk1Ver* překročí nastavenou hodnotu(5000) tak se proměnná *out* nastaví na *FALSE*, *t1* se resetuje na hodnotu 0 (časovač se vypne) a *zk1Ver* se nastaví na 5030 (z důvodu kalibrace se hodnota musela nastavit na větší hodnotu). Při spuštění opačného bloku *OmezK1Down* se proměnná *zk1Ver* zmenšuje, ruka se pohybuje dolů do doby než *zk1Ver* bude menší než -5000. Tento blok je použit jen v částech programu starající se o vertikální pohyb, horizontální pohyb nemá žádné omezení.

6.1.2 Vnitřní část bloku Motor1

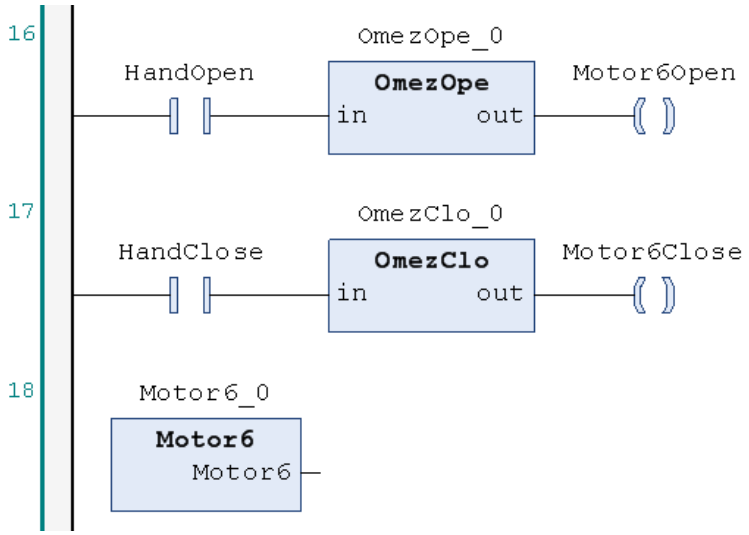


Obr. 6.3: Programový blok: Motor1

Vstupní proměnné *Motor1Up/Down* jsou ovládány přes kontakty se stejným jménem v programovém bloku *Main*. Tyto proměnné ovládají blok *Sel(Selektor)*, když je proměnná *Motor1Up* nastavena na hodnotu *TRUE*, tak je potom proměnná *a1* nastavena na hodnotu 12. V případě když je proměnná *Motor1Down* nastavena na hodnotu *TRUE*, tak je potom proměnná *b1* nastavena na hodnotu -12. Jestli tyto případy nejsou splněny tak jsou proměnné nastavené na hodnotu 0.

Poslední selektor určuje hodnotu výstupní proměnné *Motor1* a přitom zaručuje, že nenastane případ, u kterého by proměnná *Motor1* byla nastavena jak na hodnotu 12 tak -12. Kdyby nastal případ, že by *a1* bylo nastaveno na 12 a přitom *b1* bylo nastaveno na -12 tak kvůli tomuto selektoru se výstupní hodnota nastaví na 12 a ruka se začne posouvat nahoru. Stejné funkce mají i ostatní bloky *Motor* které ovládají zbývající posuny.

6.1.3 Otevírání a zavírání ruky



Obr. 6.4: Část jednotky Main zabývající se Otevíráním a zavíráním ruky

Na obrázku 6.4 lze vidět čast jednotky *Main* která se stará o otevírání a zavírání robotické ruky. Jsou zde vstupní kontakty *HandOpen* a *HandClose*, pomocí kterých se ovládá motor Motor6 přes výstupní kontakty *Motor6Open/Close*. Bloky *OmezOpe* a *OmezClo* se starají o omezení otevření a zavření ruky, pracují na stejném principu jako bloky *OmezK1Up* a *OmezK1Down* s upravením kódem.

```
IF in=TRUE
THEN
zOpC1:=zOpC1+1;
     IF zOpC1<=2500
          THEN
               out:=TRUE;
          ELSE
               out:=FALSE;
               zOpC1:=2500;
          END_IF
ELSE
     out:=FALSE;
END_IF
```

Úprava kódu byla provedena v části zabývající se omezením (z hodnoty 5000 na hodnotu 2500 => možné posunutí se zmenšilo o polovinu) a byla odstraněna část s proměnou t1, která pracovala s časovačem který v této části projektu není.

6.2 Grafické rozhraní vytvořené v jazyce C#

Grafické rozhraní pro program robotická ruka vytvoříme pomocí Visual Studia s využitím objektově orientovaného programovacího jazyka C#. Byl vyvinut firmou Microsoft, založen byl na jazycích C++ a Java. C# je navržený pro vytváření různorodých aplikací, které běží v rozhraní .NET Framework a je nejvíce využíván k tvorbě databázových programů, webových aplikací, webových služeb, formulářových aplikací ve Windows a podobných dalších. V této práci bude využit C# pro vytvoření grafického rozhraní pomocí šablony Windows Forms Application. Vytvořený program se bude skládat ze tří oken (forem), první okno bude výběrové menu z kterého je možné se přesunout do okna manuální ovládání nebo do okna automatické ovládání. Pomocí těchto ovládacích oken se ovládá program v TWinCATu.

Pro vytvoření komunikace mezi TwinCATem a Visual Studiem je potřeba přidat knihovnu TwinCAT.Ads.dll. Je to .NET knihovna z TwinCAT AdsCommLib pomocí které se mapují jednotlivé softwarové moduly (TwinCAT PLC) na třídy kompatibilní s technologií .NET, které pak mohou spolu vzájemně komunikovat.

Tab. 6.1: Přehled tříd Ads

Třída	Popis
TcAdsClient	Souhrn knihovny TcAds.dll umožňující synchronní přístup k Ads zařízením
AdsStream	Datový proud napojený na AdsCommunication
AdsBinaryWriter	Zapisuje binární hodnoty do proměnných v TwinCATu
AdsBinaryReader	Načítá jednoduché datové typy i celé struktury jako binární hodnoty

Vytvoření TcClienta a připojení k PLC:

```
TcAds[Název TcClienta] = new TcAdsClient();
TcAds.Connect("10.0.2.15.1.1"[NetID zařízení], 851[číslo portu]);
```

NetID je rozšířený identifikátor vycházející z IP adresy.

Vytvoření nového streamu:

```
AdsStream ds[název streamu] = new AdsStream(1)[délka streamu v bytech];
BinaryWriter bw[název zápisu] = new BinaryWriter(ds);
```

Zápis hodnoty do PLC:

```
ds.Position = 0[Pozice streamu];
bw.Write[Zápis]((Boolean[typ])true[hodnota která se má zapsat]);
TcAds.Write(TcAds.CreateVariableHandle("Main.Start1"), ds);
```

V prvním bodě se určí pozice streamu kam se má zapisovat, po té se do zápisu vloží hodnota která bude zaslána do TwinCATu. CreateVariableHandle vytváří handle který určuje cestu k proměnné v TwinCATu. Pomocí přikazu *TcAds.Write* se pošle hodnota, která je vložena do streamu *ds* do proměnné *Main.Start1* v TwinCATu.

```
TcAds.Dispose();
```

Příkaz ukončující spojení s Ads zařízením.

6.2.1 Výběrové menu

Při zapnutí programu se pomocí tohoto okna vybere typ ovládání. Po výběru a zmačknutí tlačítka se toto okno skryje.

```
this.Hide();
```

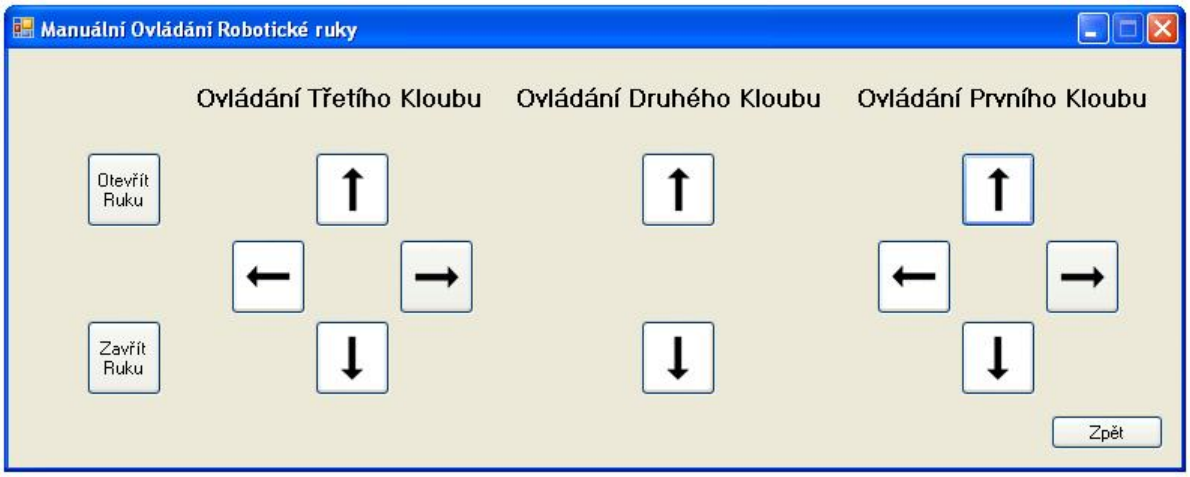
A poté se otevře okno sloužící pro ovládání TwinCATu.

```
Form1 f1 = new Form1();
f1.ShowDialog();
```

Obr. 6.5: Výběrové menu

6.2.2 Manuální ovládání

Tato část grafického rozhraní slouží pro manuální řízení simulaci robotické ruky, toto rozhraní lze vidět na obrázku 6.6. Manuální řízení slouží pro řízení ruky pomocí tlačítek (nahoru, dolů, doprava a doleva), které řídí pohyb ruky v reálném čase. Tlačítka se ovládají pomocí kurzoru kliknutím či podržením. Pro ovládání prvního kloubu (rameno) slouží 4 tlačítka (2 pro vertikální pohyb a 2 pro horizontální pohyb), druhý kloub (loket) má jen 2 tlačítka které slouží pro ovládání vertikálního pohybu a třetí kloub (zápěstí) má 4 tlačítka sloužící stejně jako u prvního kloubu. Po té jsou tu tlačítka ovládající úchop ruky a tlačítko zpět které vrací program do okna výběru ovládání obrázek 6.5.

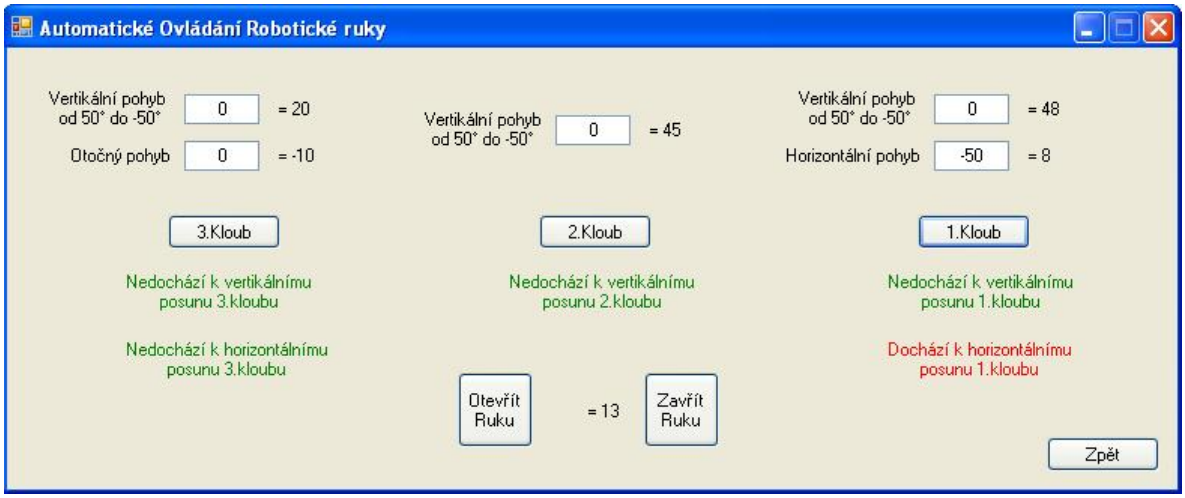


Obr. 6.6: Okno manuálního ovládání

Zmáčknutím tlačítka nahoru u prvního kloubu se spustí kontakt *Start1*, který lze vidět na obrázku 6.2, přes tento kontakt se spustí *Motor1* a výsledkem je, že se začne rameno pohybovat směrem nahoru.

6.2.3 Automatické ovládání

Okno automatického ovládání které lze vidět na obrázku 6.7 slouží pro ovládání robotické ruky jen pomocí úhlů které se zadávají do připravených textových polích. V okně jsou 3 tlačítka které ovládají příslušné klouby pomocí textových polích jsou to tlačítka *1.Kloub*, *2.Kloub* a *3.Kloub*. Tlačítka *1.Kloub* (rameno) a *3.Kloub* (loket) ovládají ruku přes dvě textová pole které se starají o vertikální a horizontální pohyb. Tlačítko *2.Kloub* (zápěstí) má jen jedno textové pole které se stará jen o vertikální pohyb. Tlačítka *Otevřít ruku* a *Zavřít ruku* pracují stejně jako se stejným názvem v manuálním módu obrázek 6.6.

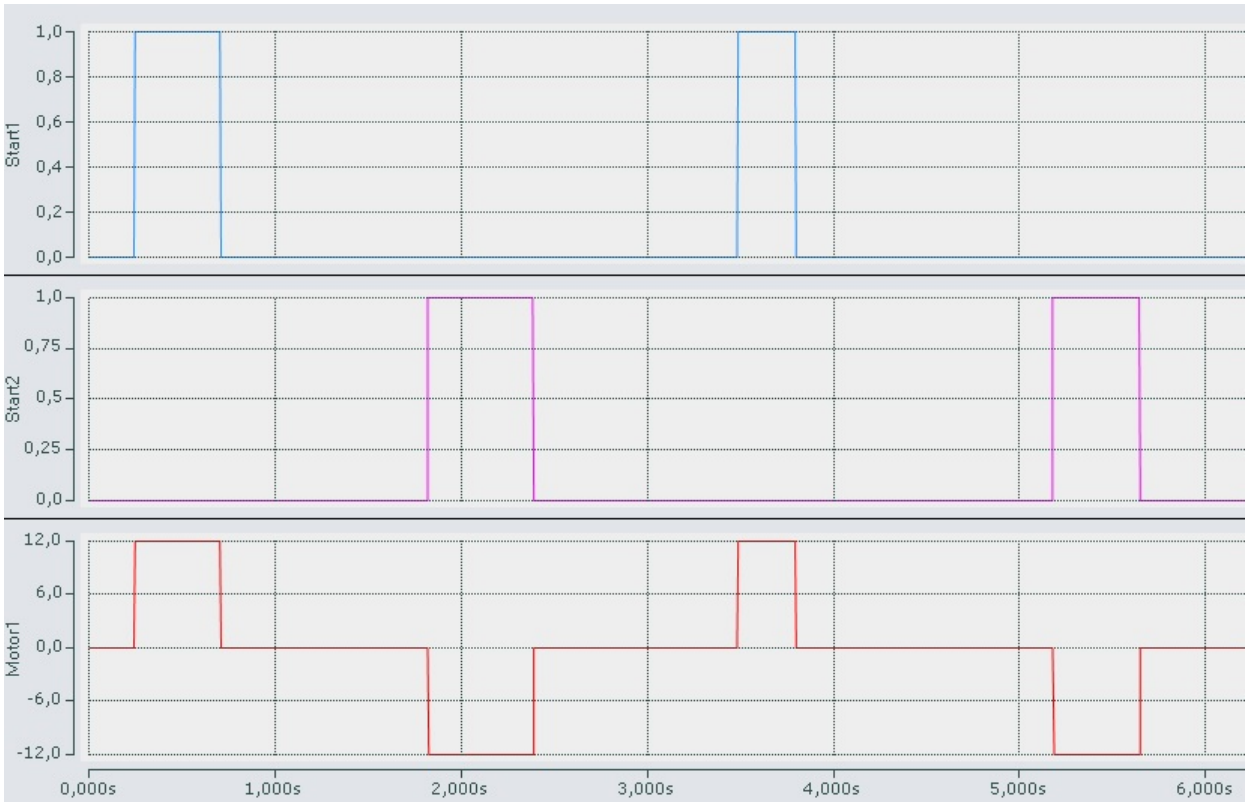


Obr. 6.7: Okno automatického ovládání

Popisky u textových polích označují posun ruky ve stupních. Při zmáčknutí tlačítka *1.Kloub* se hodnota z vertikálního textového pole pošle do proměnné *i1/2* (*i1* pro kladnou hodnotu a *i2* pro zápornou hodnotu) která lze vidět na obrázku 6.2, po úpravě se hodnota přesune do časovače který se stará o to jak dlouho se ruka bude pohybovat. Zapnutím časovače se změní text u pohybujícího se kloubu z „Nedochází k ... posunu 1.kloub" na text „Dochází k ... posunu 1.kloubu" do doby než se tento text změní nazpět, tento kloub nelze ovládat.

6.3 Výsledné grafy

6.3.1 Manuální část:

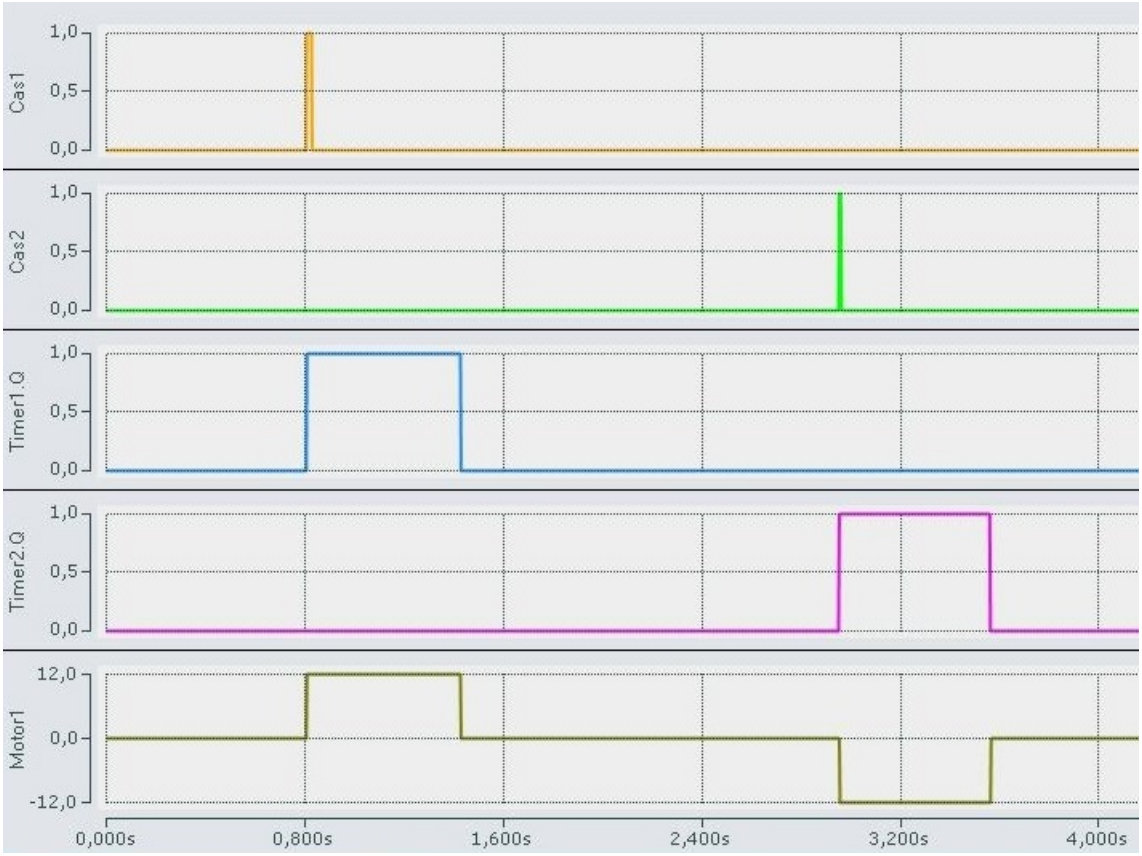


Obr. 6.8: Graf Manuální části

Popis grafu:

1. Graf Start1 – Při zmáčknutí tlačítka *Nahoru* na obrázku 6.6 se *Start1* změní na hodnotu 1 (*TRUE*) a tímto dojde k sepnutí motoru *Motor1*. Hodnota *Motor1* se změní z 0 na 12.
2. Graf Start2 – Při zmáčknutí tlačítka *Dolů* na obrázku 6.6 se *Start2* změní na hodnotu 1 (*TRUE*) a tímto dojde k sepnutí motoru *Motor1*. Hodnota *Motor1* se změní z 0 na -12.
3. Graf Motor1 – Určuje jakým směrem se ruka pohybuje při +12 se ruka pohybuje nahoru, při -12 se ruka pohybuje dolů.

6.3.2 Automatická část:



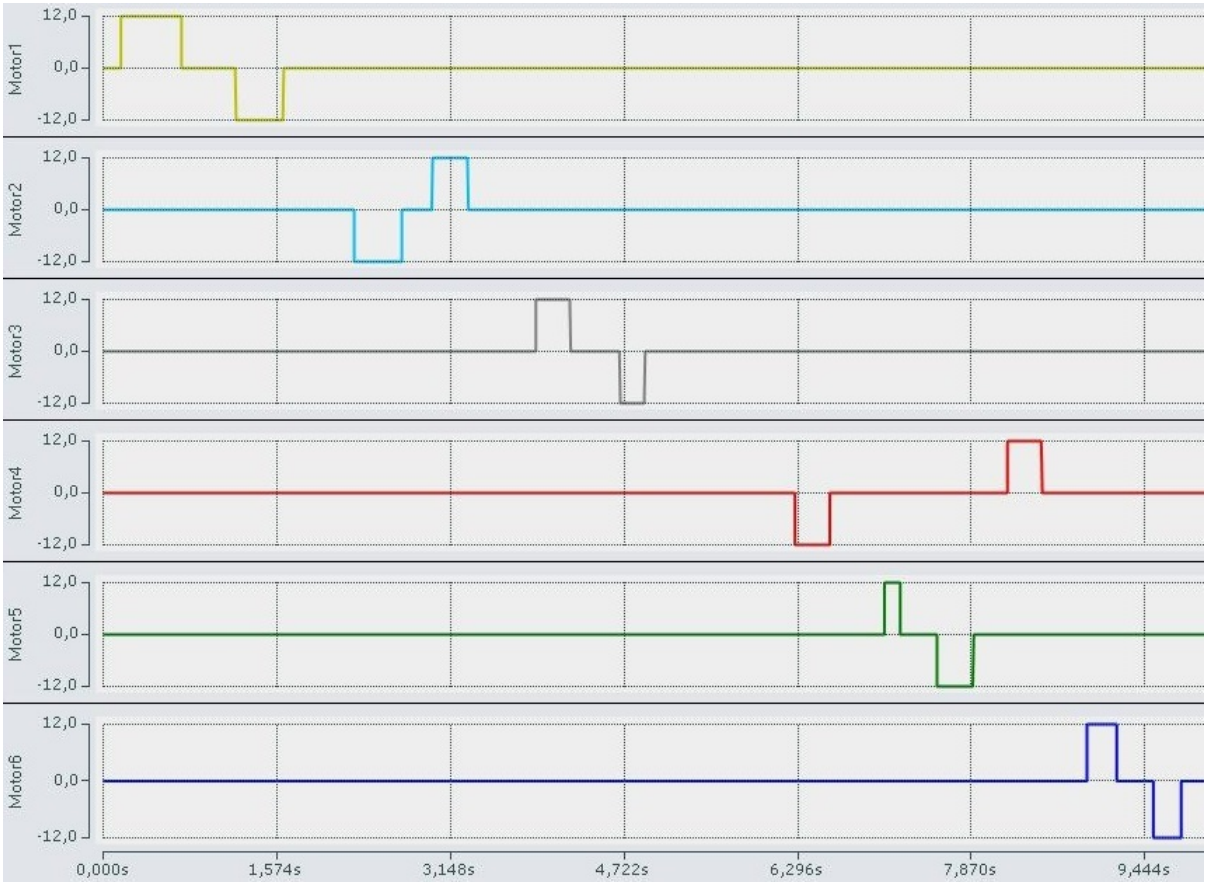
Obr. 6.9: Graf Automatické části

Popis grafu:

1. Graf Cas1 – Při zmáčknutí tlačítka *1.Kloub* na obrázku 6.7 se hodnota proměnné *Cas1* změní na hodnotu 1 (*TRUE*) a tím se zapne časovač *Timer1*.
2. Graf Cas2 – Při zmáčknutí tlačítka *1.Kloub* na obrázku 6.7 se hodnota proměnné *Cas2* změní na hodnotu 1 (*TRUE*) a tím se zapne časovač *Timer2*.
3. Graf Timer1.Q – Při zapnutí časovače se výstupní hodnota *Q* změní na hodnotu 1 (*TRUE*) do doby *t1*, po dobu spuštěného časovače je zapnutý i motor *Motor1* na hodnotě 12.
4. Graf Timer2.Q – Při zapnutí časovače se výstupní hodnota *Q* změní na hodnotu 1 (*TRUE*) do doby *t2*, po dobu spuštěného časovače je zapnutý i motor *Motor1* na hodnotě -12.
5. Graf Motor1 – Určuje jakým směrem se ruka pohybuje při +12 se ruka pohybuje nahoru, při -12 se ruka pohybuje dolů.

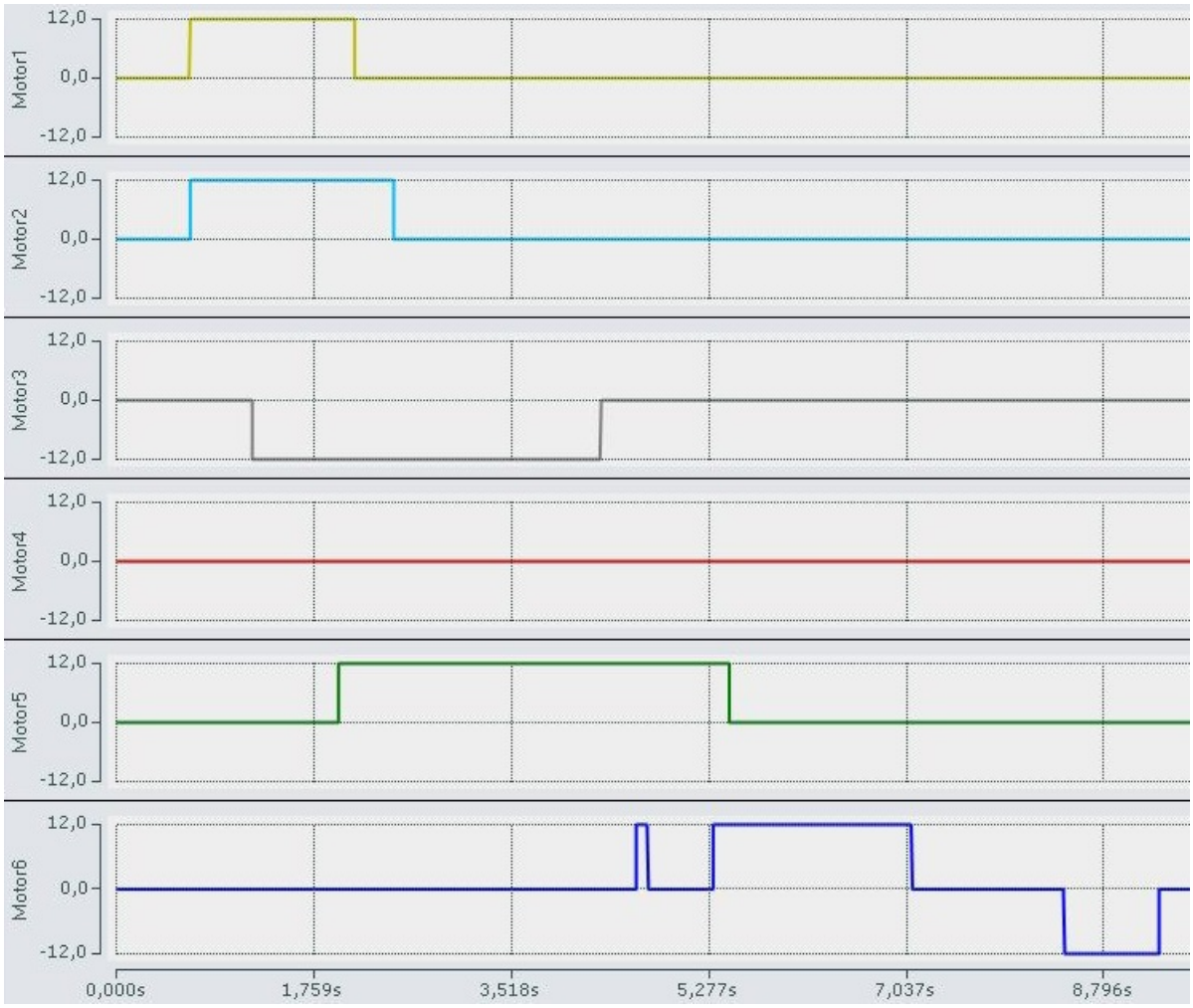
7 VÝSLEDKY SIMULACE

Výsledné grafy vycházející z vytvořeného programu robotická ruka, popisující funkci motorů.



Obr. 7.1: Graf motorů ovládaných manuální částí

Manuální část simulace se ovládá pomocí manuálního řízení 6.6. Pro ukázku 7.1 byly tlačítka na ovládání postupně mačkány a některé byly drženy déle. Graf Motor1 kontroluje motor, který se stará o posunutí 1.kloubu (rameno) vertikálním směrem při hodnotě +12 se ruka posunuje směrem nahoru, při hodnotě -12 se ruka posunuje směrem dolů. Pro 1.kloub je ještě graf Motor2 který kontroluje motor, který se stará o posunutí horizontálním směrem při hodnotě +12 se ruka posunuje směrem doprava, při hodnotě -12 se ruka posunuje směrem doleva. 2.kloubu (loket) patří graf Motor3 který kontroluje motor starající se o vertikální posunutí. 3.kloub (zápěstí) má grafy dva Motor4 (vertikální směr) a Motor5 (horizontální směr).Poslední graf Motor6 popisuje funkci motoru, který se stará o otevírání (+12) a zavírání ruky (-12).



Obr. 7.2: Graf motorů ovládaných automatickou částí

Před zapnutím simulace řízené automatickým ovládáním 6.7, byl první kloub otočen o 45 stupňů nahoru a o 20 stupňů doleva, druhý kloub byl otočen o 25 stupňů nahoru a třetí kloub byl otočen o 50 stupňů směrem dolů a o 20 stupňů doprava. Při simulaci se první kloub otočí o 10 stupňů nahoru (graf Motor1) a o 6 stupňů doprava (graf Motor2) , z grafu 7.2 lze vidět, že vertikální otočení neproběhlo kompletně, toto je způsobeno blokem omezení, který ukončí otáčení při snaze překročit 50 stupňů (viz kapitola 6.1.1). Druhý kloub při simulaci se otočí o 10 stupňů dolů (graf Motor3). Třetí kloub se otočí o 10 stupňů doprava (graf Motor5) a měl se otočit i o 10 stupňů dolů (graf Motor4), toto otočení neproběhlo, protože kloub už byl otočen na jeho maximální polohu (50 stupňů). Z posledního grafu lze vyčíst,že ruka byla napřed otevřena a pote zase zavřena,

7.1 Diskuze výsledku

Tuto práci lze využít pro simulaci robotických ruk, při oddělání nebo naopak při přidání dalších motorů lze simulovat jednodušší (méně pohyblivých kloubu) nebo složitější (více pohyblivých kloubu) robotické ruky. Změnou handlu (kapitola 6.2) v grafickém rozhraní lze po té, toto rozhraní využít pro ovládání jiných robotických ruk a posuvných robotů, které pracují s PLC naprogramovaným pomocí TwinCATu.

Práci je také možné upravit pro fungování v reálných podmínkách, po této úpravě by bylo možné ovládat reálnou robotickou ruku s výše vytvořeným grafickým rozhraním.

8 ZÁVĚR

Práce se zabývala simulací v TwinCATu. Cílem práce bylo seznámit se s programovým logickým automatem, jeho funkcí a současným využitím. Dále pak byl popsán vývojový nastroj pro PLC od společnosti BECKHOFF TwinCAT.

V rámci práce byly popsány programovací jazyky vývojového nástroje TwinCAT ve verzi 2, které tvoří celkový program. Poté zde byly popsány systémové informace TwinCATu 2 a typy sběrnic přes které TwinCAT komunikuje z programovým logickým automatem.

Přínosem práce byla simulace PLC pomocí TwinCATu. V rámci této simulace byly popsány jednotlivé jednotky a bloky v nich vytvořené. Program vytvořený pro tuto simulaci má základ v již vytvořeném programu od BECKHOFFu. Tento program vytvářel vrtačku která po 25 krocích vrtala díru do obrobku, s úpravou programu vznikla vrtačka která se nejen posouvala vertikálně ale také horizontálně, byl také snížen počet kroků než se dá vrtačka do pohybu z 25 na 20 kroků. Důvod vzniklé úpravy byl kvůli obrobku u kterého by nestačil jen vertikální posun. Dále je v práci popsána simulace jednotlivých jednotek a bloků, co se děje s ostatními jednotkami když pracuje blok co se stará o posun vrtačky doleva. Popis jednotlivých operátorů a proměnných jak v bloku Init, tak i záložce globálních proměnných. Na konci práce je pak graf simulace vytvořený v programu TwinCAT, který ukazuje kdy a jak dlouho jednotlivé funkce pracují. Graf také ukazuje proměnou kroky, která ukazuje počet kroků a když dosáhne hodnoty 20, tak se vypne motor a začne pracovat vrtačka.

Hlavním přínosem práce bylo vytvoření simulace robotické ruky za pomoci programu TwinCAT ve verzi 3, který se po instalaci spojí s Visual studiem, takto vytvořené spojení z jednodušší práci při které by se musel tvořit ještě program v C++/C# nebo v jiném programovacím jazyku který podporuje Visual Studio. Tohoto zjednodušení se hned využilo při vytvoření grafického rozhraní v programovém jazyce C#. Pomocí knihovny TwinCAT.Ads.dll a tříd v této knihovně bylo možné komunikovat s programem v PLC. Z důvodu simulace nebyly řešeny problémy které by mohly nastat v reálném použití (váha ramena, váha předmětu který by se zkoušel zvednout, překážek v trase ruky).

Program robotická ruka byl naprogramován tak, aby ho bylo možné ovládat dvěma způsoby pomocí manuálního ovládání a automatického ovládání. Obě metody ovládání jsou naprogramované v jednom programu C# a mezi nimi se dá jednoduše přepínat. Manuální ovládání je nastaveno tak, že při zmačknutí či přidržení tlačítka se dá ruka do pohybu. Automatické ovládání pracuje na principu zadávání stupňů, které se potom pošlou do TwinCATu tam se převedou na čas který určuje jak dlouho se má ruka posunovat určeným směrem (protisměry se určují znaménkem před za-

daným úhlem). Manuální a automatické ovládání je na závěr vykresleno do grafů. V poslední části je pak řešena výsledná simulace a výsledky jsou poté vykresleny do grafů.

LITERATURA

[1] Samin, R.E., Lee Ming Jie a Zawawi, M.A. *PID implementation of heating tank in mini automation plant using Programmable Logic Controller (PLC)*. IEEE. Fac. of Electr. & Electron. Eng., Univ. Malaysia Pahang, Pekan, Malaysia: Pahang, 2011n. l. ISBN 978-1-61284-229-5. Dostupné také z: http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=5953937

[2] Kabalan, M., Tamir, D. a Singh, P. *Electrical load controller for rural micro-hydroelectric systems using a programmable logic controller*. IEEE. Electr. & Comput. Eng., Villanova Univ., Villanova, PA, USA: Ottawa, ON, 2015. ISBN 978-1-4799-8961-4. Dostupné také z: http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=7238042

[3] Mahmood, O.T. *Programmable logic controller based design and implementation of multiple axes solar tracking system*. IEEE. Dept. of Electr. Technol., Tech. Inst. / Hawija, Kirkuk, Iraq: Mosul, 2013. ISBN 978-1-4799-5633-3. Dostupné také z: http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6998742

[4] Seidel Schlenker, C., Diaz, J.J., Morales, L.F., Ciendua, L.A., Patino, A.A. a Montoya, J.O. *Design and implementation of a machine for the production of potato chips*. IEEE. Programa de Ing. en Automatizacion, Univ. de La Salle, Bogota, Colombia: Bogota, 2013. ISBN 978-1-4799-2470-7.

[5] EHRIG, Hartmut (ed.), Gabriel JUHAS (ed.), Julia PADBERG (ed.) a Grzegorz ROZENBERG (ed.). *Unifying Petri Nets: Advances in Petri Nets*. Springer. Berlin: Springer-Verlag Berlin Heidelberg, 2001. ISBN 978-3-540-43067-4.

[6] SMEU, G.A. *Automatic conveyor belt driving and sorting using SIEMENS step 7-200 programmable logic controller*. IEEE. Bucharest, Romania: Politeh. of Bucharest, 2013. ISBN 978-1-4673-5979-5. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6563408

[7] BURHAN, I., S. TALIB a A.A. AZMAN. *Design and fabrication of Programmable Logic Controller Kit with multiple output module for teaching and learning purposes*. IEEE. Aircraft Maintenance Eng. Dept., Politek. Sultan Salahuddin Abdul Aziz Shah, Shah Alam, Malaysia: Melaka, 2012. ISBN 978-1-4673-0960-8. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6194681

[8] ASHOUR, Hamdy. *Automatic transfer switch (ATS) using programmable logic controller (PLC)*. IEEE. Dept. of Electr. & Comput. Control Eng., Arab Acad. for Sci. & Technol., Alexandria, Egypt: Dept. of Electr. & Comput. Control Eng., Arab Acad. for Sci. & Technol, 2004. ISBN 0-7803-8599-3. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1364495

[9] BINDER, B.J.T., Kufoalor D.K.M, Pavlov A a Johansen T.A. *Embedded Model Predictive Control for an Electric Submersible Pump on a Programmable Logic Controller*. IEEE. Trondheim, Norsko: Juan Les Antibes, 2014. ISBN 978-1-4799-7407-8. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6981402

[10] YAHYAEI, M. a A.W. LABIB. *Increasing the Flexibility and Intelligence of Material Handling through the Factory by Integrated Fuzzy Logic Controller with Programmable Logic Controller*. IEEE. Reno, NV: Manchester Univ., 2005. ISBN 0-7803-9159-4. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1452428

[11] ZHAO, Futao, Wei DU, Yiheng XU a Zhiren HU. *Programmable logic controller applied in steam generators water levels*. IEEE. San Diego, CA: Inst. of Instrum. in Chem. Eng., Zhejiang Univ., Hangzhou, China, 1996. ISBN 0-7803-3544-9. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=559273

[12] TSUKAMOTO, T. a K. TAKAHASHI. *Modeling of elevator control logic based on Mark Flow Graph and its implementation on programmable logic controller*. IEEE. Tokyo: Dept. of Electr. & Electron. Eng., Tokyo Inst. of Technol., Tokyo, Japan, 2014. ISBN 978-1-4799-5146-8. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7031225

[13] ALHERAISH, A., W. ALOMAR a M. ABU-AL-ELA. *Programmable Logic Controller System for Controlling and Monitoring Home Application Using Mobile Network*. IEEE. Sorrento: Dept. of Electr. Eng., King Saud Univ., Riyadh, 2006. ISBN 0-7803-9359-7. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4124369

[14] PO-JEN, Cheng, Cheng CHIN-HSING a Chang TZAI-SHIANG. *Variant-frequency fuzzy controller for air conditioning driver by programmable logic controller*. IEEE. Kaohsiung: Dept. of Electr. Eng., Nan Jeon Inst. of Technol., Tainan Hsien, Taiwan, 2011. ISBN 978-89-956056-4-6. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5899236

[15] BESADA-PORTAS, E., J.A LOPEZ-OROZCO, L DE LA TORRE a J.M. DE LA CRUZ. *Remote Control Laboratory Using EJS Applets and TwinCAT Programmable Logic Controllers*. IEEE. Španělsko: Comput. Archit. & Autom. Dept., Univ. Complutense of Madrid, Madrid, Spain, 2012. ISBN 0018-9359. Dostupné také z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6220875

[16] LYDON, Bill (ed.). BECKHOFF. *Automation.com* [online]. 67 Alexander Drive: Automation.com, 2013, 2013-09-06 [cit. 2015-11-17]. Dostupné z: http://www.automation.com/library/case-studies/iec-61131-3-based-building-automation-system

[17] VOJÁČEK, Antonín. *Automatizace.hw* [online]. 2011, 3. 3. 2011 [cit. 2015-12-13]. Dostupné z: http://automatizace.hw.cz/programovaci-rezimy-pro-plc-dle-iec-611313-codesys

[18] Beckhoff Automation. *TC3 ScopeView* [online]. [cit. 2016-04-28]. Dostupné z: https://www.beckhoff.com/english.asp?twincat/twincat-3-xa-scope.htm?id=1893323819865450

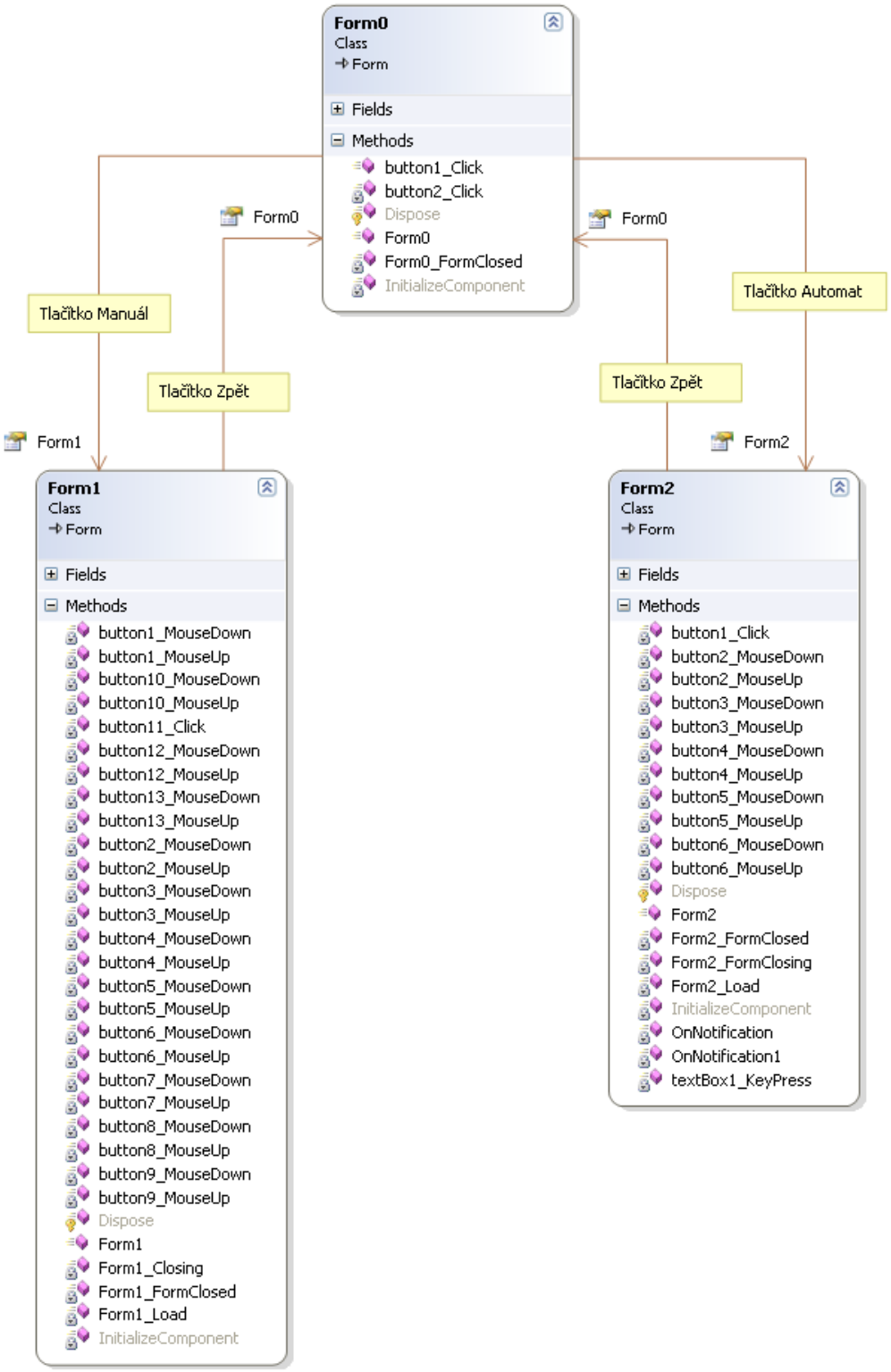
SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

PLC	Programovatelný logický automat – Programmable Logic Controller
IEC	Mezinárodní elektrotechnická komise – International Electrotechnical Commission
HMI	Rozhraní mezi člověkem a strojem – Human Machine Interface
ESP	Elektronický stabilizační program – Electronic Stability Program
SEPTIC	Statoiluv in-house software nastroj pro model prediktivní regulace – Statoil's in-house software tool for Model Predictive Control
MFG	Značený vývojový graf – Mark Flow Graph
SFC	Sekvenční funkční diagram – Sequential function chart
ATS	Automatický přenosový přepínač – Automatic Transfer Switch
IFLPLC	Integrovaný logický fuzzy kontrolér s programovatelným logickým automatem – Integrated Fuzzy Logic controller with Programmable Logic Controller
FLC	Logický fuzzy kontrolér– Fuzzy Logic Controller
GSM	Globální Systém pro Mobilní komunikaci – Global System for Mobile Communications
SMS	Služba krátkých textových zpráv – Short Message Service)
EJS	Jednoduchá Java simulace – Easy Java Simulations
CPU	Centrální procesorovou jednotku – Central Processing Unit
IPC	Průmyslové PC – Industrial PC
IL	Jazyk seznamu instrukcí – Instruction List
ST	Jazyk strukturovaného textu – Structured Text
LD	Jazyk příčkového diagramu – Ladder Diagram
FBD	Jazyk funkčního blokového schématu – Function block diagram
CFC	Jazyk volně propojovaných bloků – Continuous Function Chart
SFC	Sekvenční funkční diagram – Sequential Function Chart

POU	Programová organizační jednotka – Program Organisation Unit
TOF	Časovač zpoždění vypnutí – Timer Off-Delay
SEL	Binární Selektor – Binary Selection

SEZNAM PŘÍLOH

A PŘILOŽENÝ DIAGRAM TŘÍD



Obr. A.1: Diagram tříd popisující program grafického rozhraní